June 21, 2021
TAP 2021, Online

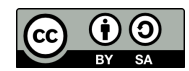
A Benchmarks Library for Extended Parametric Timed Automata

Étienne André[1], Dylan Marinho[1] and Jaco van de Pol[2]

[1] Université de Lorraine, CNRS, Inria, LORIA, Nancy, France

[2] Department of Computer Science, Aarhus University, Aarhus, Denmark

Supported by the ANR-NRF French-Singaporean research program ProMiS (ANR-19-CE25-0015)



Motivation

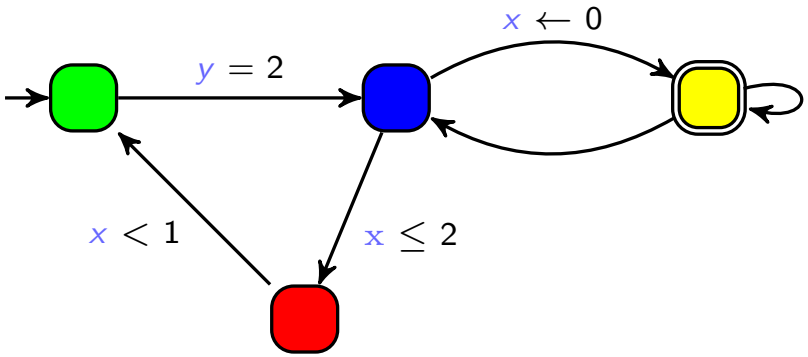
- Real-time systems:
  - Systems for which not only the functional correctness but also the time answer is important

Motivation

- Critical Real-time systems:
  - Systems for which not only the functional correctness but also the time answer is important
  - Failures (in correctness or timing) may result in dramatic consequences

Outline

Timed model checking

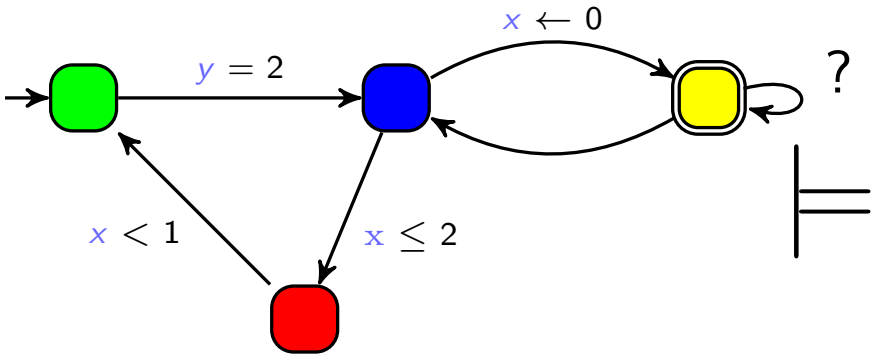


A model of the system

is unreachable

A property to be satisfied

Timed model checking



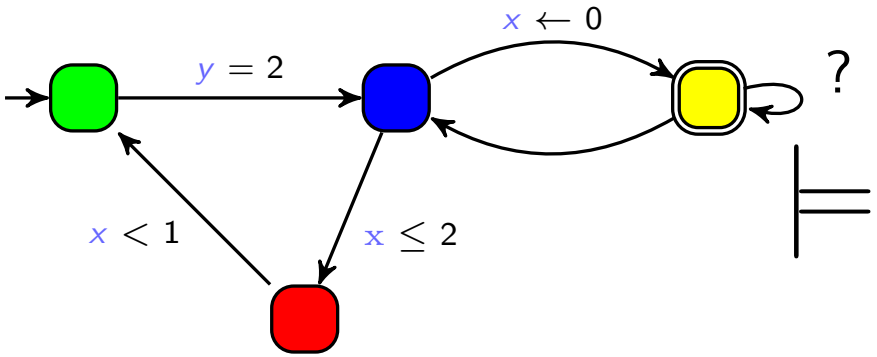
A model of the system

is unreachable

A property to be satisfied

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

Timed model checking



A model of the system

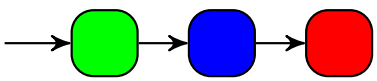
is unreachable

A property to be satisfied

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

Yes

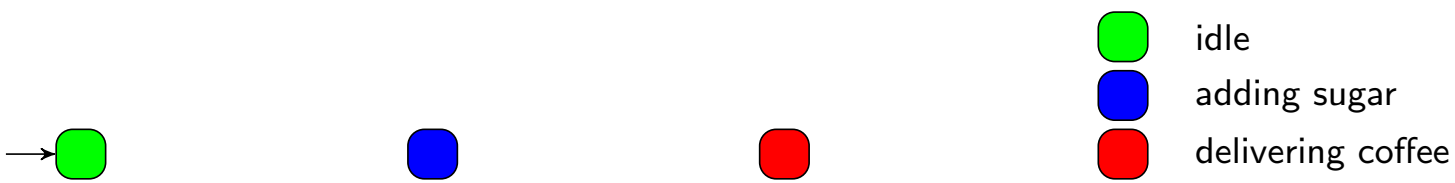
No



Counterexample

Timed automaton (TA)

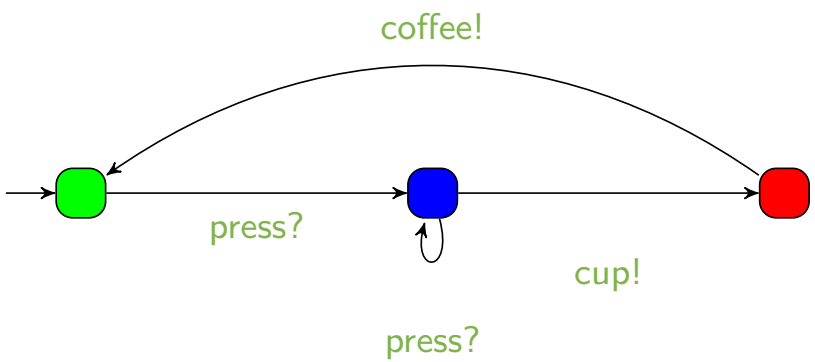
- Finite state automaton (sets of locations)

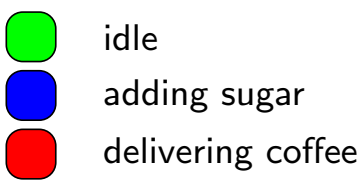


[AD94] Rajeev Alur and David L. Dill. "A theory of timed automata". In: *Theoretical Computer Science* 126.2 (Apr. 1994), pp. 183–235. ISSN: 0304-3975

Timed automaton (TA)

- Finite state automaton (sets of locations and actions)

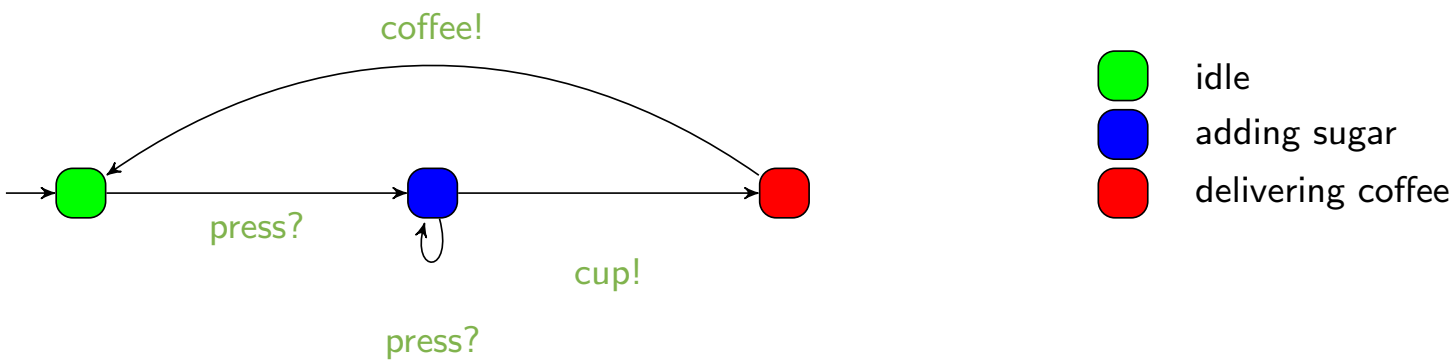




[AD94] Rajeev Alur and David L. Dill. "A theory of timed automata". In: *Theoretical Computer Science* 126.2 (Apr. 1994), pp. 183–235. ISSN: 0304-3975

Timed automaton (TA)

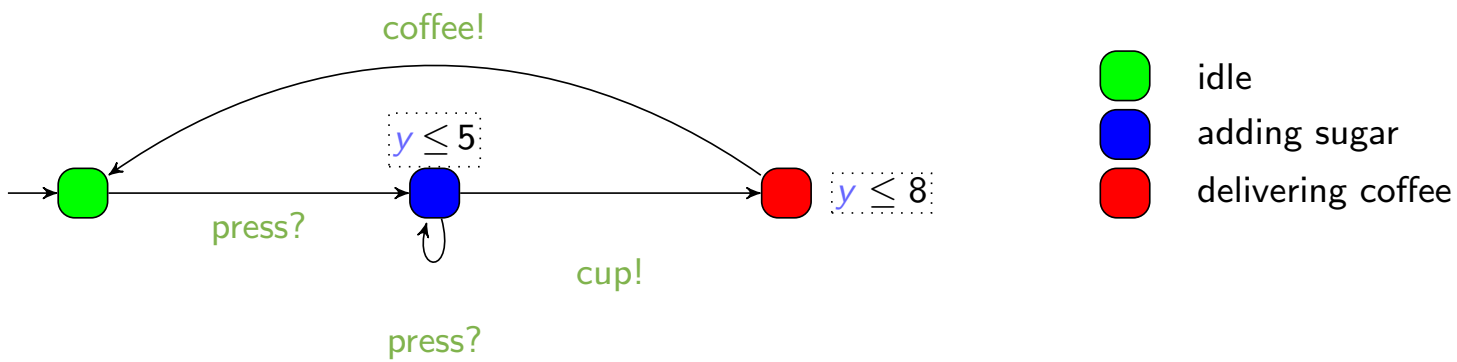
- Finite state automaton (sets of locations and actions) augmented with a set X of clocks [AD94]
  - Real-valued variables evolving linearly at the same rate
  - Can be compared to integer constants



[AD94] Rajeev Alur and David L. Dill. "A theory of timed automata". In: *Theoretical Computer Science* 126.2 (Apr. 1994), pp. 183–235. ISSN: 0304-3975

Timed automaton (TA)

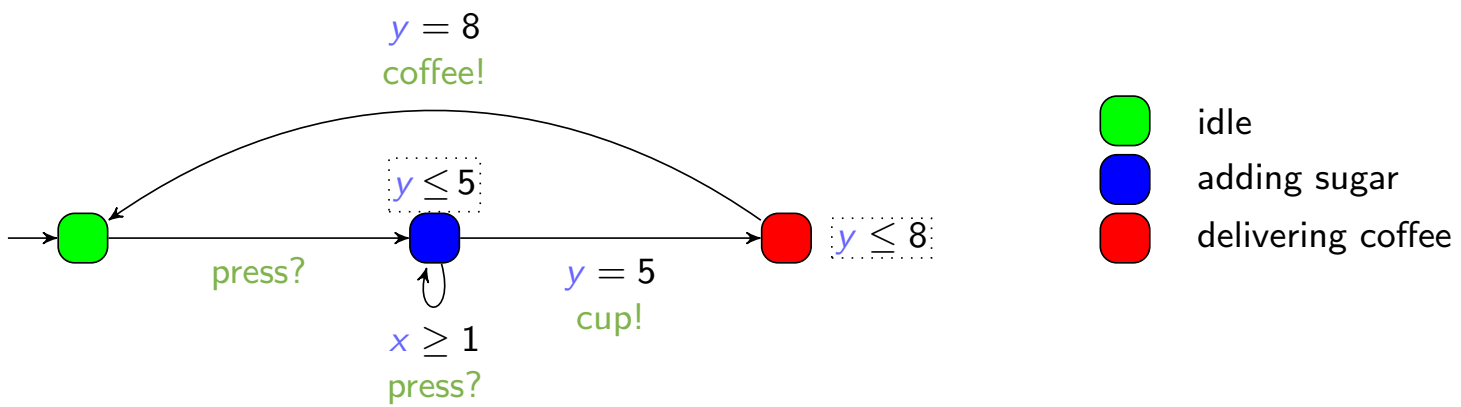
- Finite state automaton (sets of locations and actions) augmented with a set X of clocks [AD94]
  - Real-valued variables evolving linearly at the same rate
  - Can be compared to integer constants in invariants
- Features
  - Location invariant: property to be verified to stay at a location



[AD94] Rajeev Alur and David L. Dill. "A theory of timed automata". In: *Theoretical Computer Science* 126.2 (Apr. 1994), pp. 183–235. ISSN: 0304-3975

Timed automaton (TA)

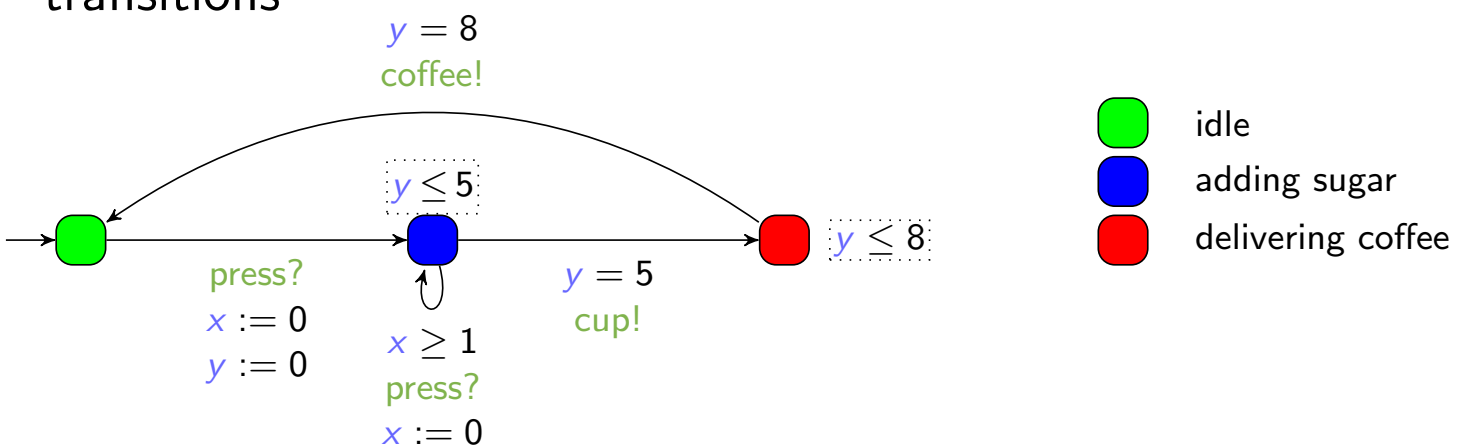
- Finite state automaton (sets of locations and actions) augmented with a set X of clocks [AD94]
  - Real-valued variables evolving linearly at the same rate
  - Can be compared to integer constants in invariants and guards
- Features
  - Location invariant: property to be verified to stay at a location
  - Transition guard: property to be verified to enable a transition



[AD94] Rajeev Alur and David L. Dill. "A theory of timed automata". In: *Theoretical Computer Science* 126.2 (Apr. 1994), pp. 183–235. ISSN: 0304-3975

Timed automaton (TA)

- Finite state automaton (sets of locations and actions) augmented with a set X of clocks [AD94]
  - 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



[AD94] Rajeev Alur and David L. Dill. "A theory of timed automata". In: *Theoretical Computer Science* 126.2 (Apr. 1994), pp. 183–235. ISSN: 0304-3975

Timed automata: a useful formalism for verification and testing

- Timed automata are a common formalism to reason about systems involving timing and concurrency
- Nice decidability results [AD94]
- Many interesting applications
  - Notably testing [Lut+17] and monitoring [AHW18]

[AD94] Rajeev Alur and David L. Dill. "A theory of timed automata". In: *Theoretical Computer Science* 126.2 (Apr. 1994), pp. 183–235. ISSN: 0304-3975

[Lut+17] Lars Luthmann et al. "Modeling and Testing Product Lines with Unbounded Parametric Real-Time Constraints". In: *SPLC, Volume A*. ACM, 2017, pp. 104–113

[AHW18] Étienne André, Ichiro Hasuo, and Masaki Waga. "Offline timed pattern matching under uncertainty". In: *ICECCS*. IEEE Computer Society, 2018, pp. 10–20

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 a given constant while preserving correctness?
  - Robustness [BMS13]: 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?

[BMS13] Patricia Bouyer, Nicolas Markey, and Ocan Sankur. "Robustness in timed automata". In: *RP*. vol. 8169. Lecture Notes in Computer Science. Invited paper. Springer, Sept. 2013, pp. 1–18

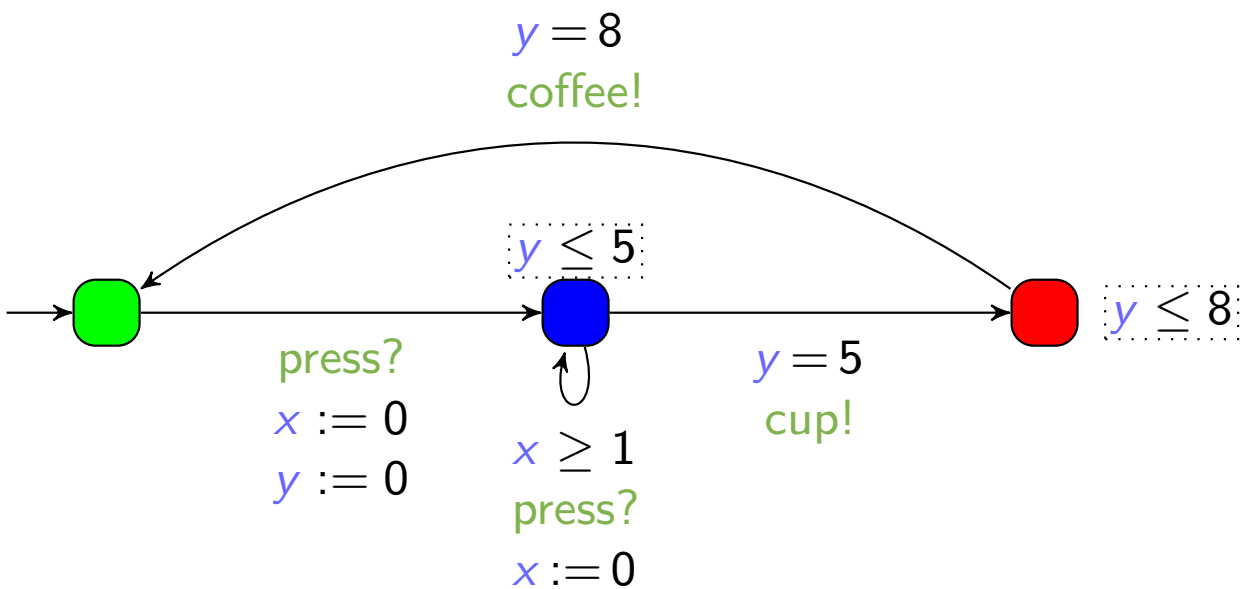
Beyond timed model checking: parameter synthesis

- 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 a given constant while preserving correctness?
  - Robustness [BMS13]: 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)

[BMS13] Patricia Bouyer, Nicolas Markey, and Ocan Sankur. "Robustness in timed automata". In: *RP*. vol. 8169. Lecture Notes in Computer Science. Invited paper. Springer, Sept. 2013, pp. 1–18

Parametric Timed Automaton (PTA)

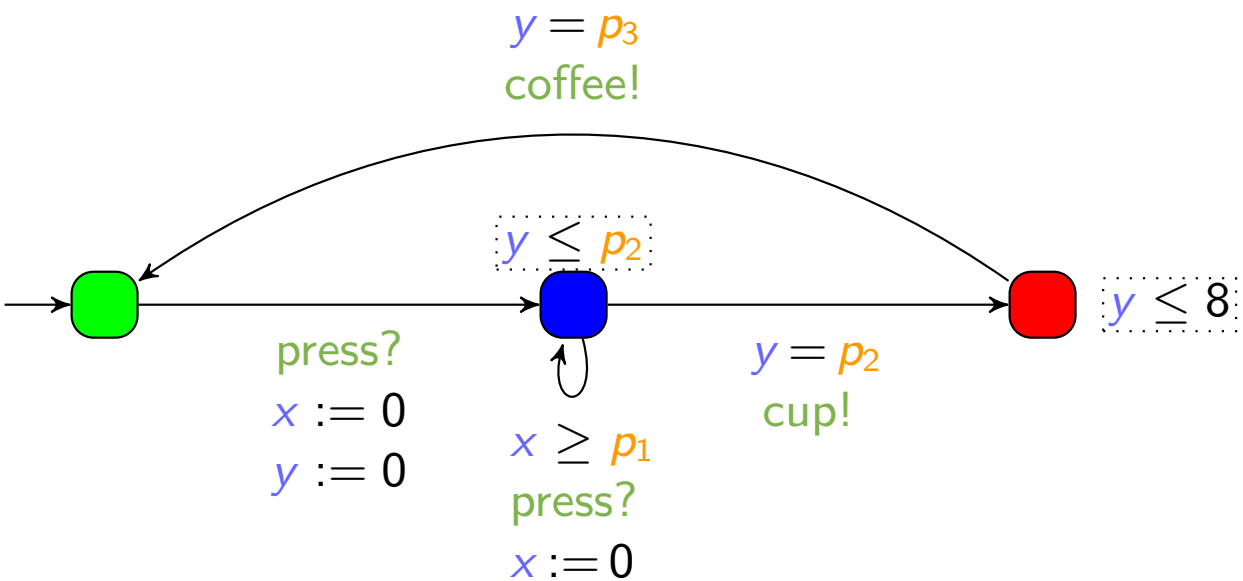
- Timed automaton (sets of locations, actions and clocks)



[AHV93] Rajeev Alur, Thomas A. Henzinger, and Moshe Y. Vardi. "Parametric real-time reasoning". In: *STOC*. ACM, 1993, pp. 592–601. ISBN: 0-89791-591-7

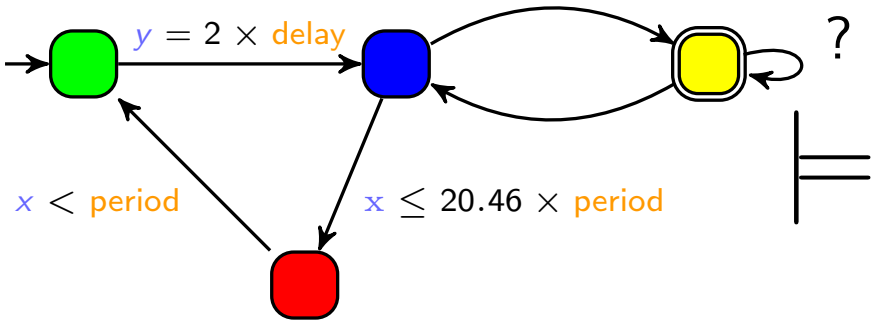
Parametric Timed Automaton (PTA)

- Timed automaton (sets of locations, actions and clocks) augmented with a set P of parameters [AHV93]
  - Unknown rational constants compared to a clock in guards and invariants



[AHV93] Rajeev Alur, Thomas A. Henzinger, and Moshe Y. Vardi. "Parametric real-time reasoning". In: *STOC*. ACM, 1993, pp. 592–601. ISBN: 0-89791-591-7

timed model checking



A model of the system

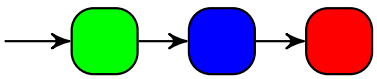
is unreachable

A property to be satisfied

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

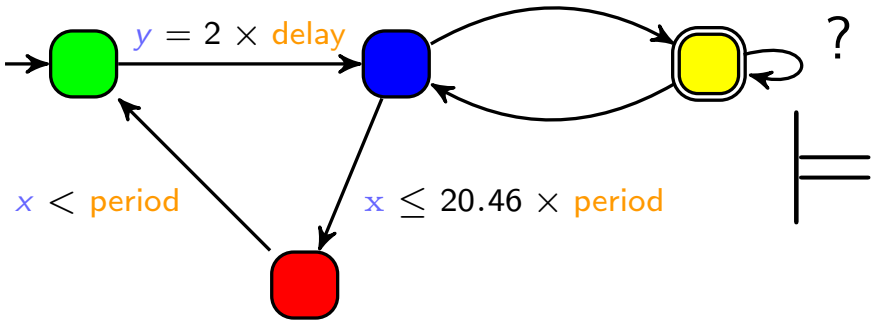
Yes

No



Counterexample

Parametric timed model checking



A model of the system

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...

$2 \times \text{delay} > 20.46 \times \text{period}$

An undecidable formalism

- Parametric timed automata are very expressive
  - Most interesting problems are undecidable [AHV93]
  - Exact parameter synthesis is out of reach
    - . . . but approximated algorithms, or exact algorithms without guarantee of termination, can be proposed
  - In practice: many interesting problems can still be solved (exactly)

[AHV93] Rajeev Alur, Thomas A. Henzinger, and Moshe Y. Vardi. "Parametric real-time reasoning". In: *STOC*. ACM, 1993, pp. 592–601. ISBN: 0-89791-591-7

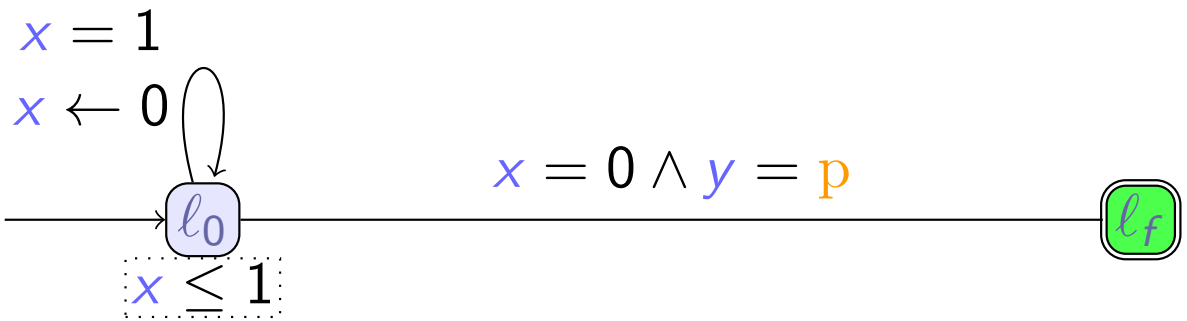
Outline

Motivation

- Test new algorithms for TAs or PTAs and check their efficiency by comparing them with existing techniques
- Improve existing techniques and tools
- Provide a set of "unsolvable" (yet very simple) benchmarks:
  - Emphasize the limits of state-of-the-art algorithms to encourage the community to address these cases

Motivation

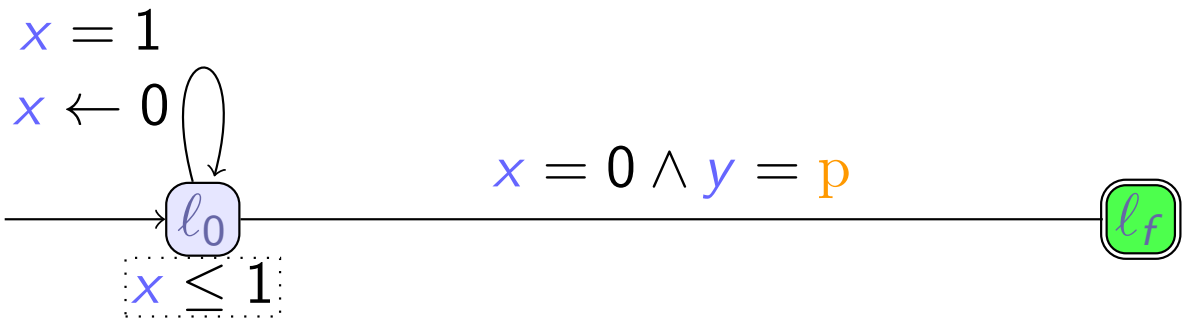
Example of an "unsolvable" model



ℓ_f is reachable iff $\mathrm{p} \in \mathbb{N}$

Motivation

Example of an "unsolvable" model

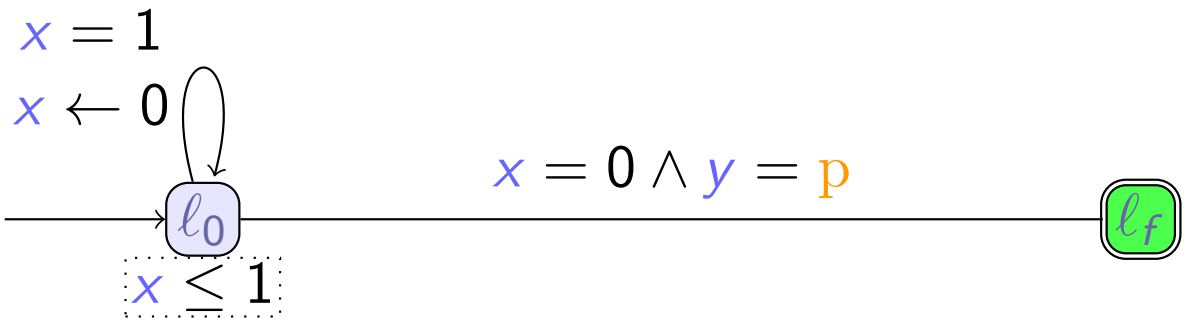


ℓ_f is reachable iff $\mathrm{p} \in \mathbb{N}$

$p = 1$:

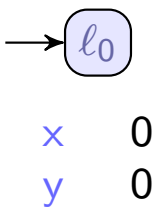
Motivation

Example of an "unsolvable" model



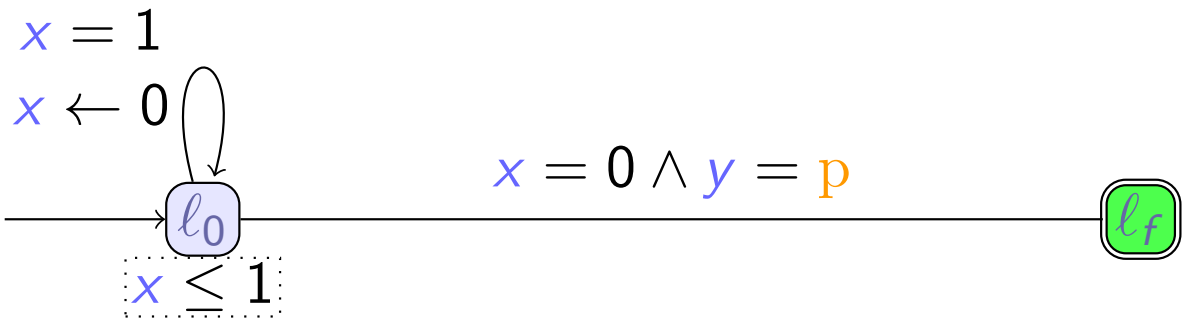
ℓ_f is reachable iff $\mathrm{p} \in \mathbb{N}$

$p = 1$:



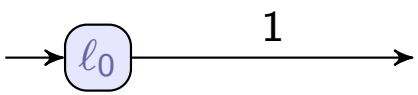
Motivation

Example of an "unsolvable" model



ℓ_f is reachable iff $\mathrm{p} \in \mathbb{N}$

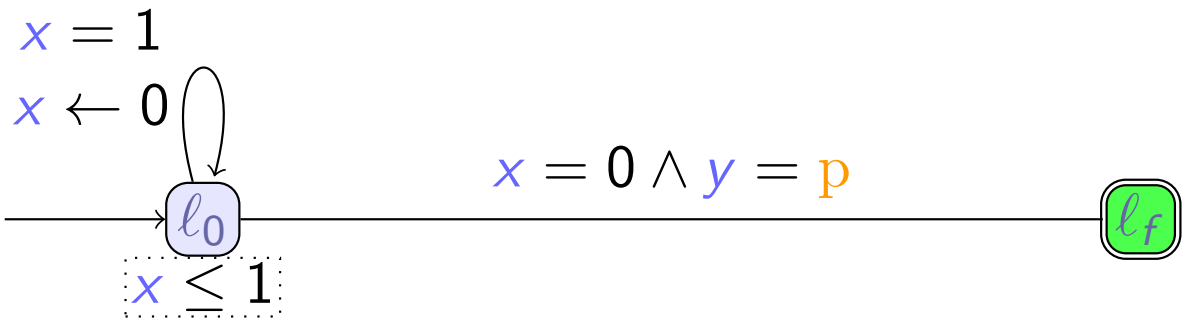
$p = 1$:



x	0
y	0

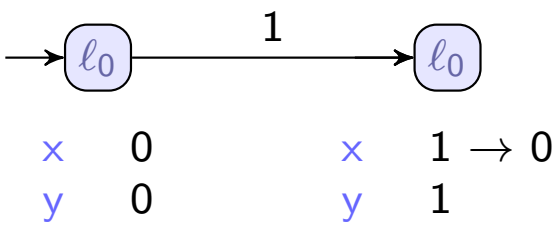
Motivation

Example of an "unsolvable" model



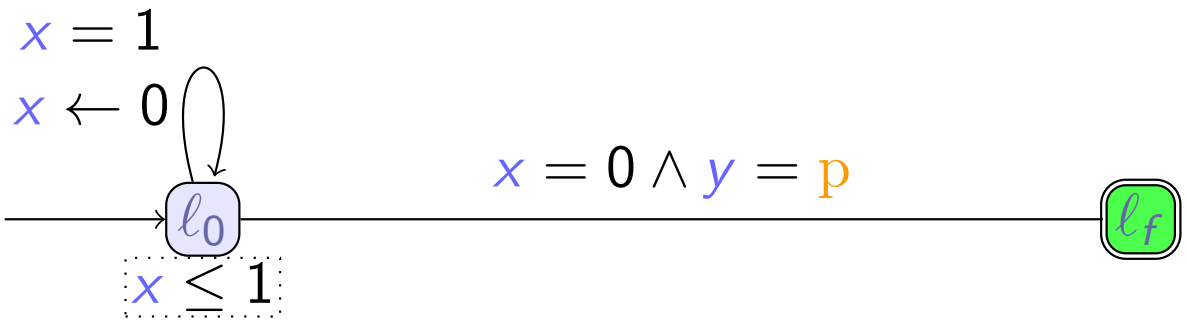
ℓ_f is reachable iff $\mathrm{p} \in \mathbb{N}$

$p = 1$:



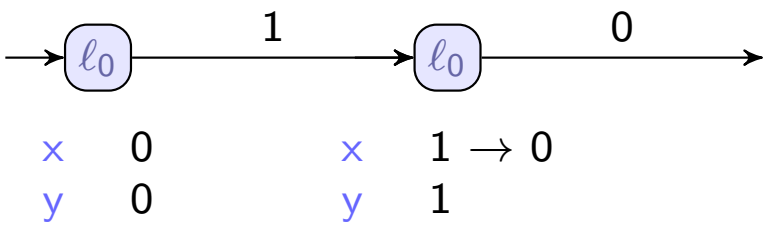
Motivation

Example of an "unsolvable" model



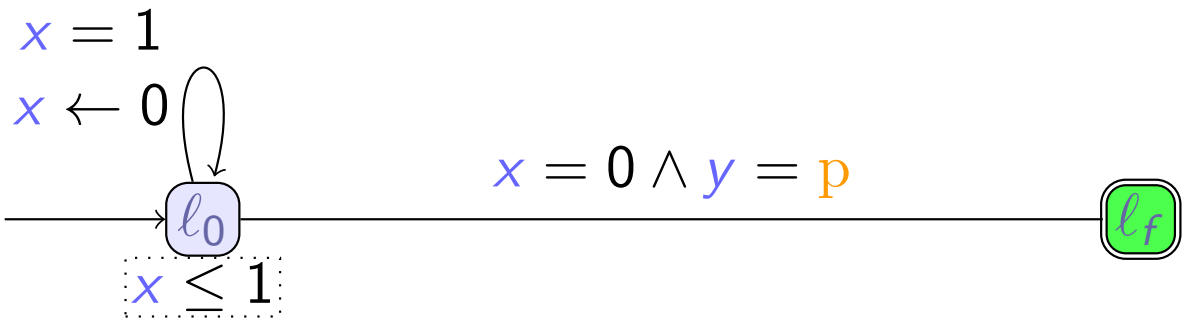
ℓ_f is reachable iff $\mathrm{p} \in \mathbb{N}$

$p = 1$:



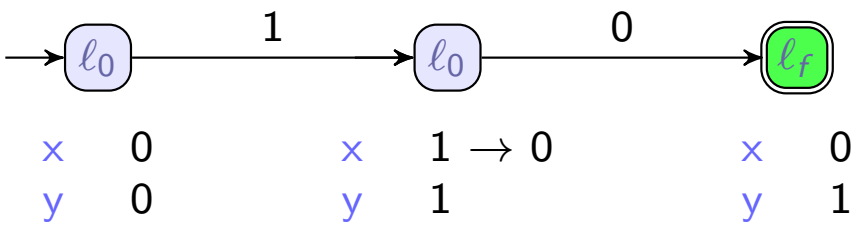
Motivation

Example of an "unsolvable" model



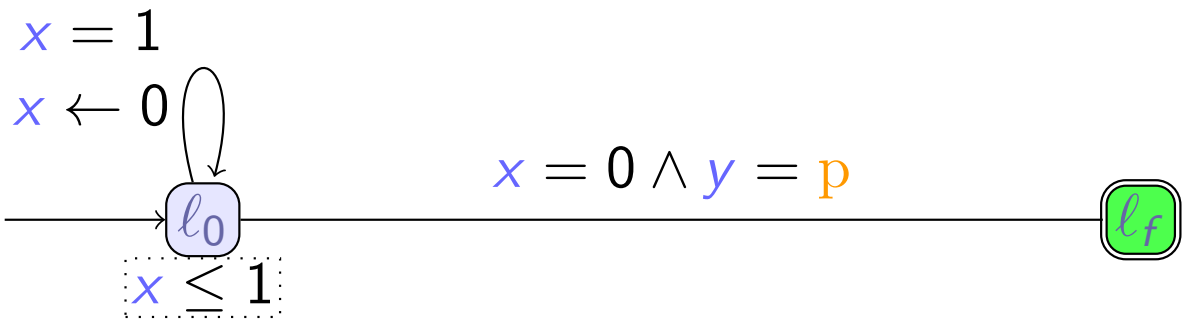
ℓ_f is reachable iff $\mathrm{p} \in \mathbb{N}$

$p = 1$:



Motivation

Example of an "unsolvable" model

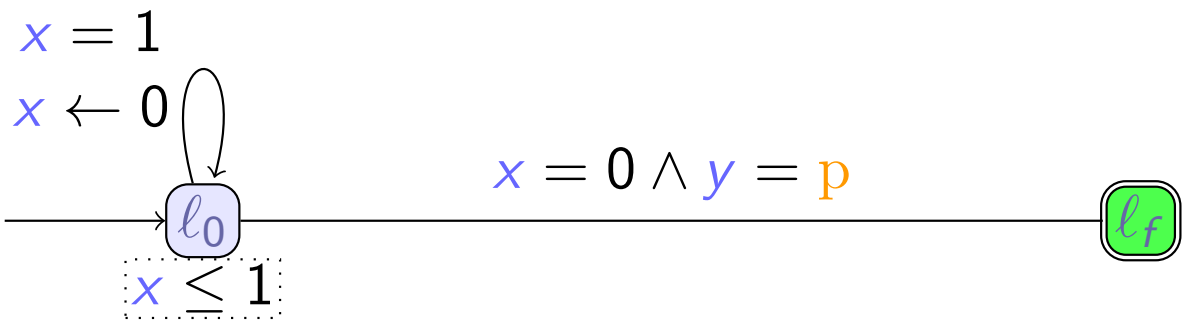


ℓ_f is reachable iff $\mathrm{p} \in \mathbb{N}$

$p = 2$:

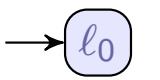
Motivation

Example of an "unsolvable" model



ℓ_f is reachable iff $\mathrm{p} \in \mathbb{N}$

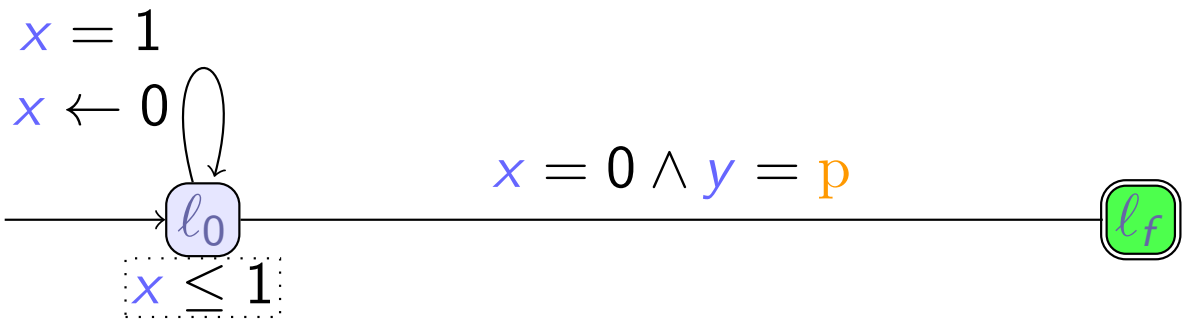
$p = 2$:



x	0
y	0

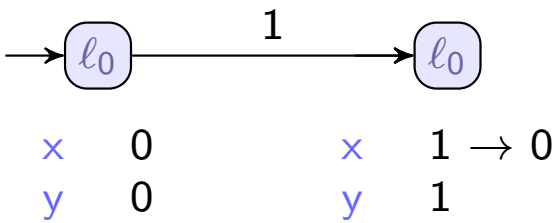
Motivation

Example of an "unsolvable" model



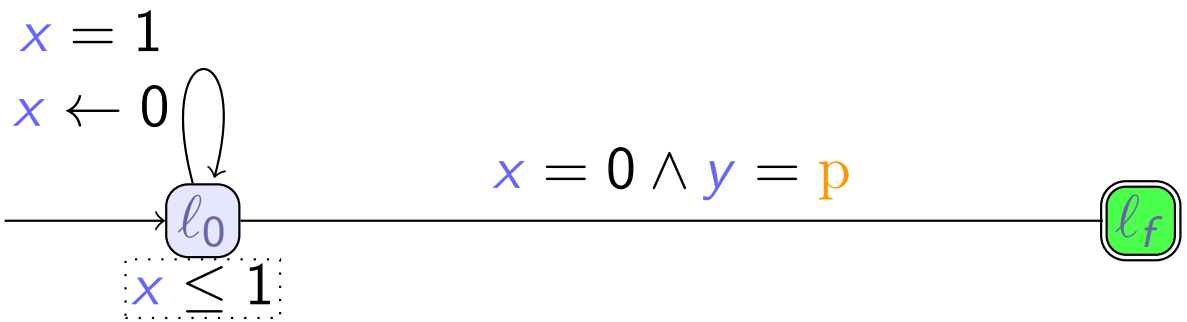
ℓ_f is reachable iff $\mathrm{p} \in \mathbb{N}$

$p = 2$:



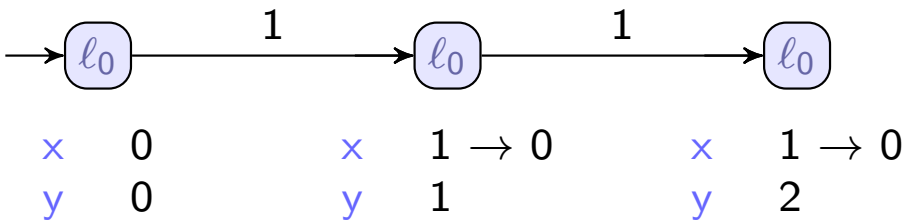
Motivation

Example of an "unsolvable" model



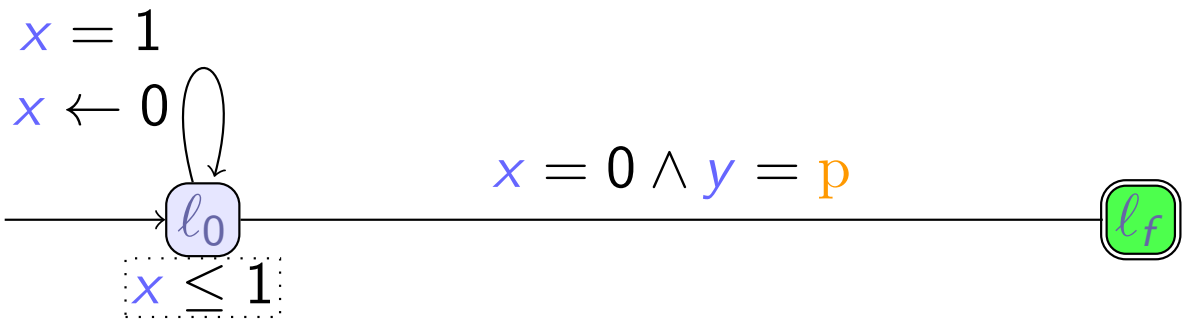
ℓ_f is reachable iff $\mathrm{p} \in \mathbb{N}$

$p = 2$:



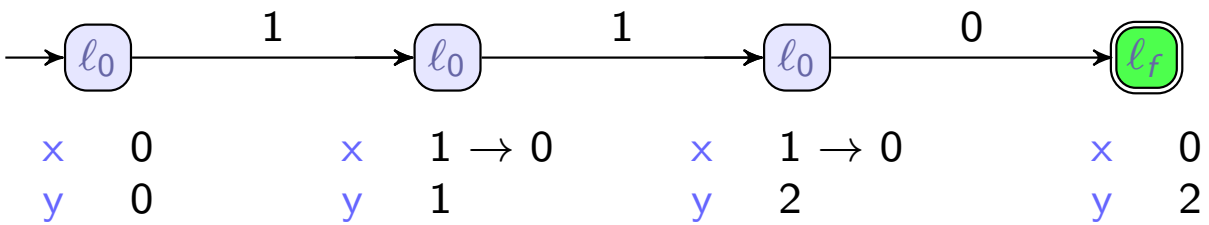
Motivation

Example of an "unsolvable" model



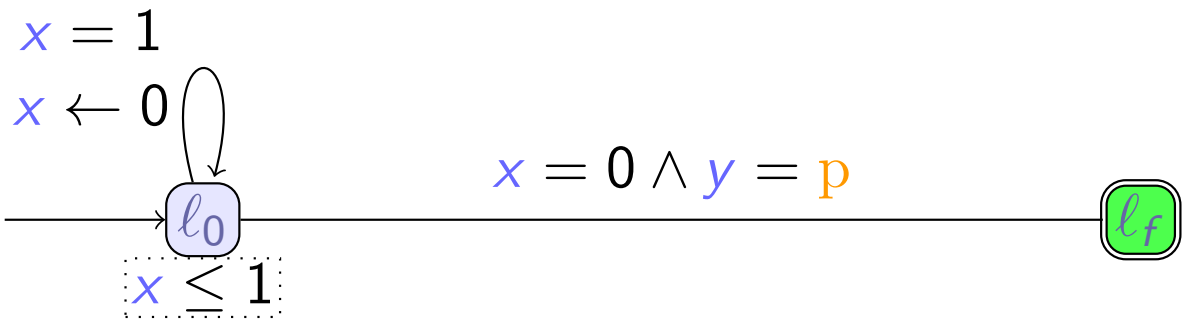
ℓ_f is reachable iff $\mathrm{p} \in \mathbb{N}$

$p = 2$:



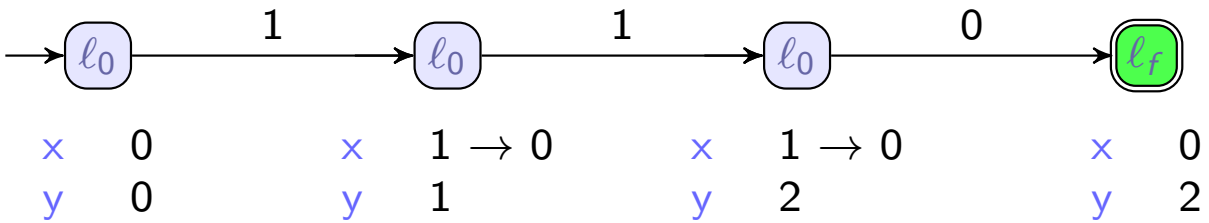
Motivation

Example of an "unsolvable" model



ℓ_f is reachable iff $\mathrm{p} \in \mathbb{N}$

$p = 2$:



Exact synthesis (i. e., inferring $\mathrm{p} = i, i \in \mathbb{N}$) is not solvable by any existing parametric timed model checker (to the best of our knowledge)

Why a new version?

- A first library [And18] exists, but
  - the syntax is only compatible with the previous version of IMITATOR, and the former calling paradigm
  - it contains exclusively safety/reachability properties
  - only syntactic information is provided
    - No expected result

[And18] Étienne André. *The IMITATOR benchmarks library v1.0*. 2018. URL: https://www.imitator.fr/library1.html

Why a new version?

- A first library [And18] exists, but
  - the syntax is only compatible with the previous version of IMITATOR, and the former calling paradigm
  - it contains exclusively safety/reachability properties
  - only syntactic information is provided
    - No expected result
- New version:
  - More benchmarks
  - Focus on liveness properties
  - Focus on unsolvable benchmarks
  - Semantic information (computation time, expected result...)

[And18] Étienne André. *The IMITATOR benchmarks library v1.0*. 2018. URL: https://www.imitator.fr/library1.html

The IMITATOR library v2

Library	Size			Metrics		Properties		
Vers.	Bench.	Models	Prop.	Static	Semantic	EF	TPS	liveness
1.0	34	80	122	√	×	√	√	×
2.0 [AMP21b]	56	119	216	√	√	√	√	√

Library	Format		Categories	Analysis
Vers.	`.imi`	Jani [Bud+17]	Unsolvable	Results
1.0	2.12	×	×	×
2.0	3.0	√	√	√

[AMP21b] Étienne André, Dylan Marinho, and Jaco van de Pol. *The IMITATOR benchmarks library v2.0*. 2021. URL: https://www.imitator.fr/static/library2/

[Bud+17] Carlos E. Budde et al. *JANI specification*. 2017. URL: https://jani-spec.org/

Origins and Provided formats

- Benchmarks come from industrial collaborations, from academic papers (from different communities), and from our experience in the field (toys, unsolvable)

- Two formats are provided:
  - The IMITATOR 3 syntax (`.imi`) [And21]
  - The `JANI Specification` translation [Bud+17]
    - A new interchange format for automata-based formalisms

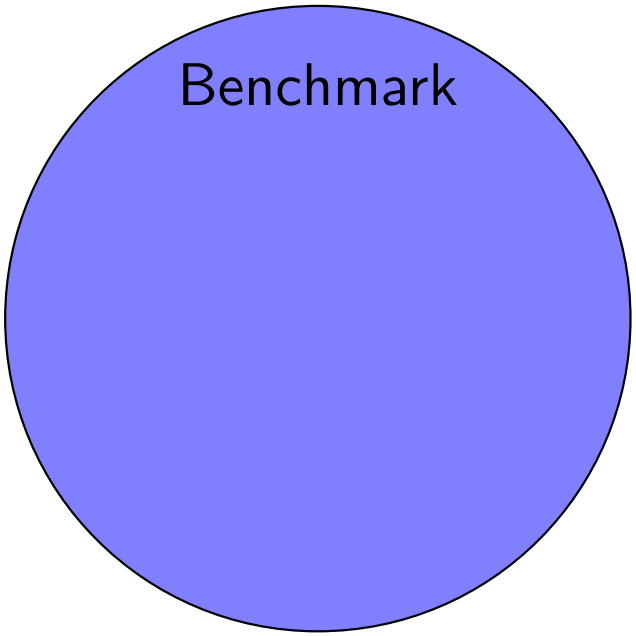
[And21] Étienne André. "IMITATOR 3: Synthesis of timing parameters beyond decidability". In: *CAV*. 2021

[Bud+17] Carlos E. Budde et al. *JANI specification*. 2017. URL: https://jani-spec.org/

Tagging our models

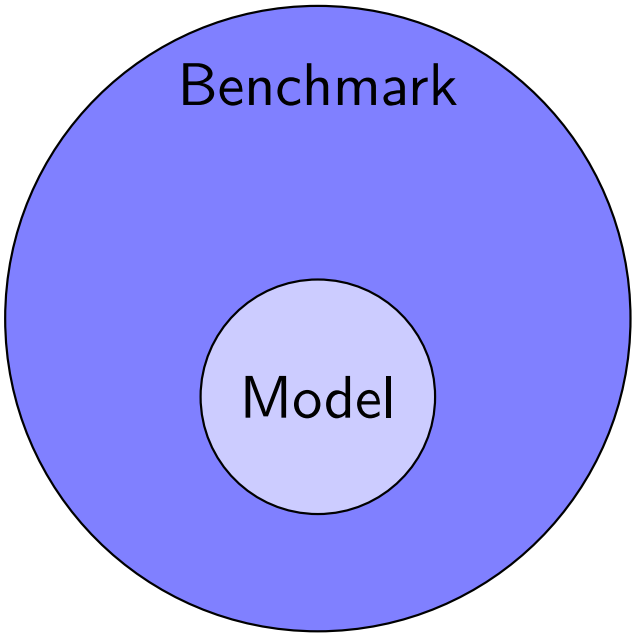
- All the benchmarks are tagged with one or more categories
- So far:
  - Academic,
  - Automotive,
  - Education,
  - Hardware,
  - Industrial,
  - Monitoring,
  - Producer-consumer,
  - Protocol,
  - Real-time system,
  - Scheduling,
  - Toy,
  - Unsolvable.

Structure of the library



Benchmark `FischerPS08`

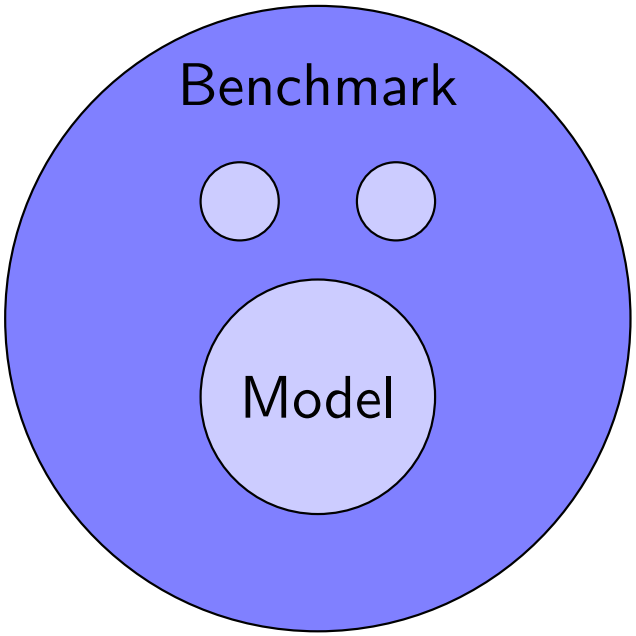
Structure of the library



Benchmark `FischerPS08`

Model `FischerPS08:2`

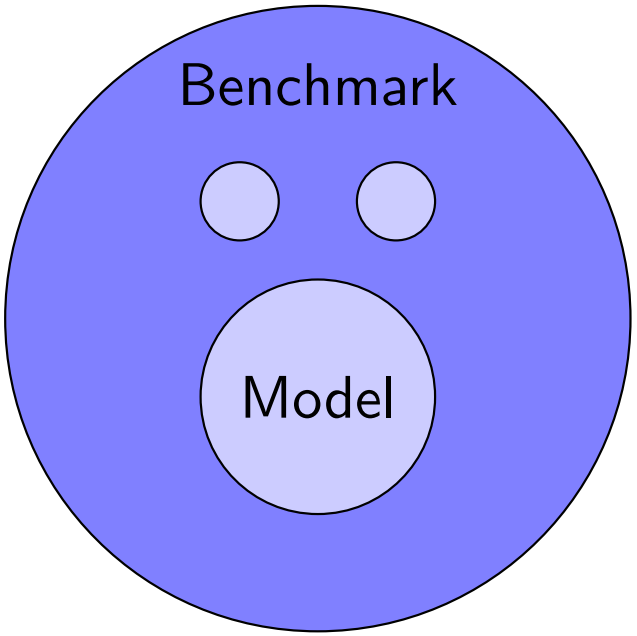
Structure of the library



Benchmark `FischerPS08`

Models `FischerPS08:2`, `FischerPS08:3`, ...

Structure of the library



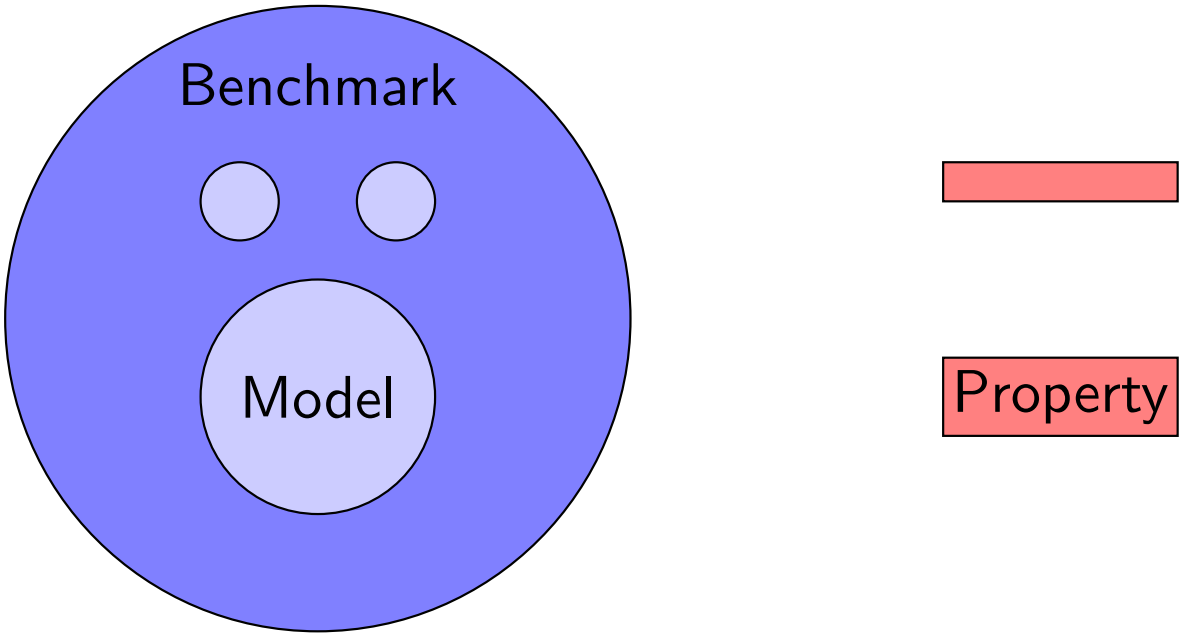
Property

Benchmark `FischerPS08`

Models `FischerPS08:2`, `FischerPS08:3`, ...

Property `FischerPS08:AGnot`

Structure of the library

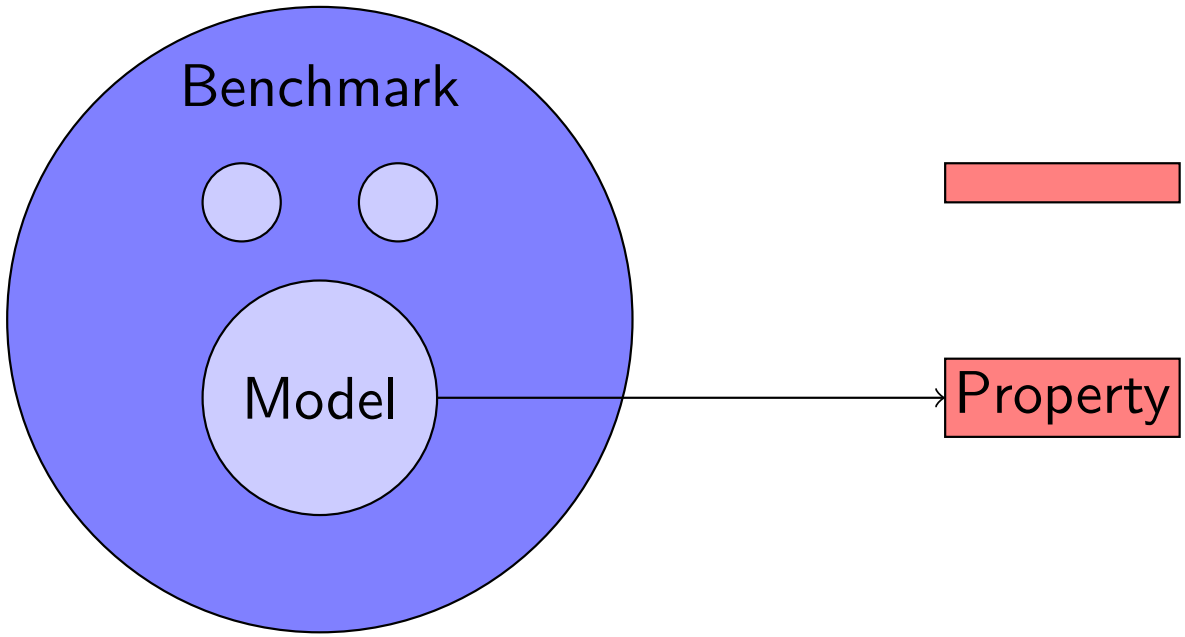


Benchmark `FischerPS08`

Models `FischerPS08:2`, `FischerPS08:3`, ...

Properties `FischerPS08:AGnot`, `FischerPS08:IM`, ...

Structure of the library

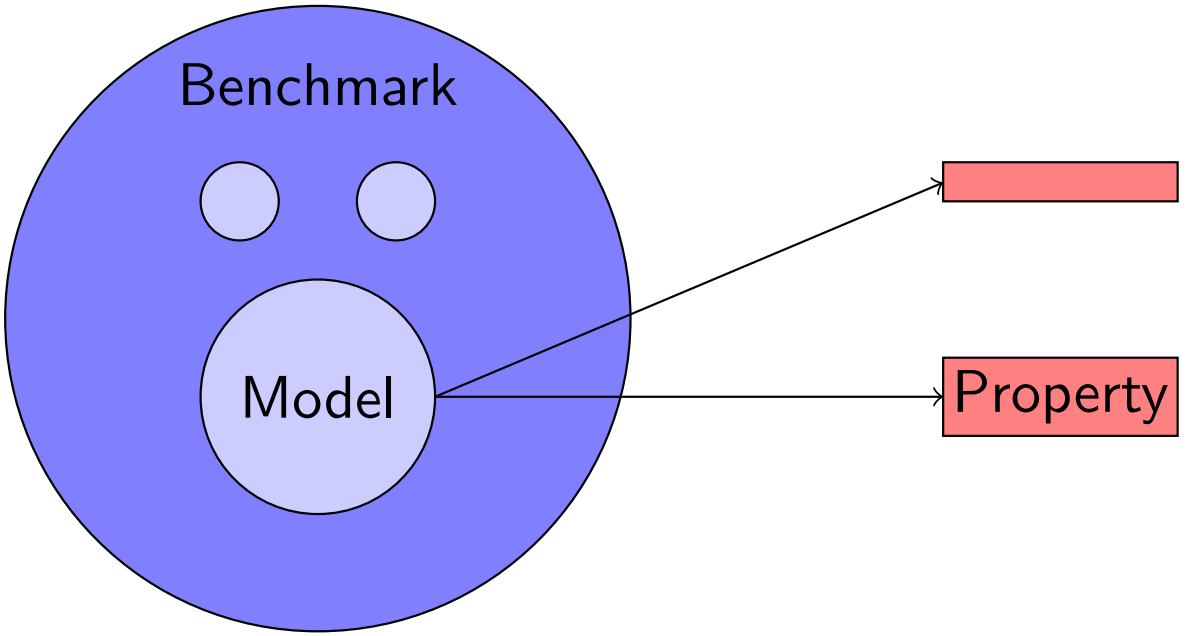


Benchmark `FischerPS08`

Models `FischerPS08:2`, `FischerPS08:3`, ...

Properties `FischerPS08:AGnot`, `FischerPS08:IM`, ...

Structure of the library

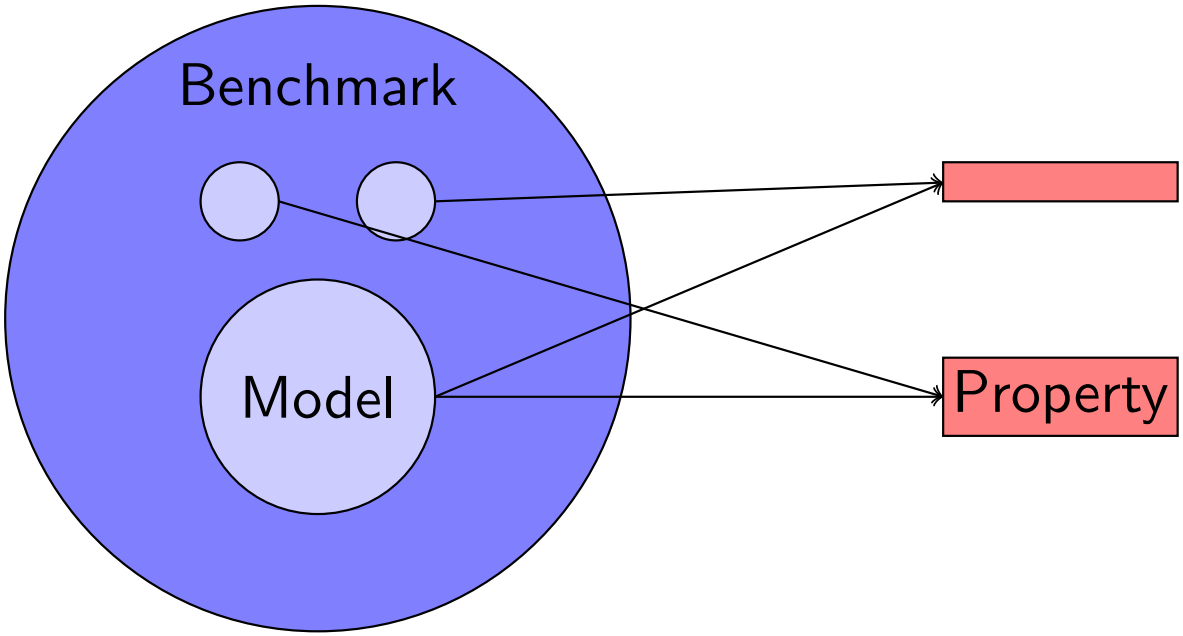


Benchmark `FischerPS08`

Models `FischerPS08:2`, `FischerPS08:3`, ...

Properties `FischerPS08:AGnot`, `FischerPS08:IM`, ...

Structure of the library



Benchmark `FischerPS08`

Models `FischerPS08:2`, `FischerPS08:3`, ...

Properties `FischerPS08:AGnot`, `FischerPS08:IM`, ...

How to select a subset of our benchmarks?

Classification	Static	Semantic
categorization	number of clocks	size of the state space
generation method	silent actions?	total computation time
scalability	...	...

How to select a subset of our benchmarks?

Classification	Static	Semantic
categorization	number of clocks	size of the state space
generation method	silent actions?	total computation time
scalability	...	...

For example

Get all benchmarks

- that are scalable,

How to select a subset of our benchmarks?

Classification	Static	Semantic
categorization	number of clocks	size of the state space
generation method	silent actions?	total computation time
scalability	...	...

For example

Get all benchmarks

- that are scalable,
- that feature no invariant nor unobservable actions

How to select a subset of our benchmarks?

Classification	Static	Semantic
categorization	number of clocks	size of the state space
generation method	silent actions?	total computation time
scalability	...	...

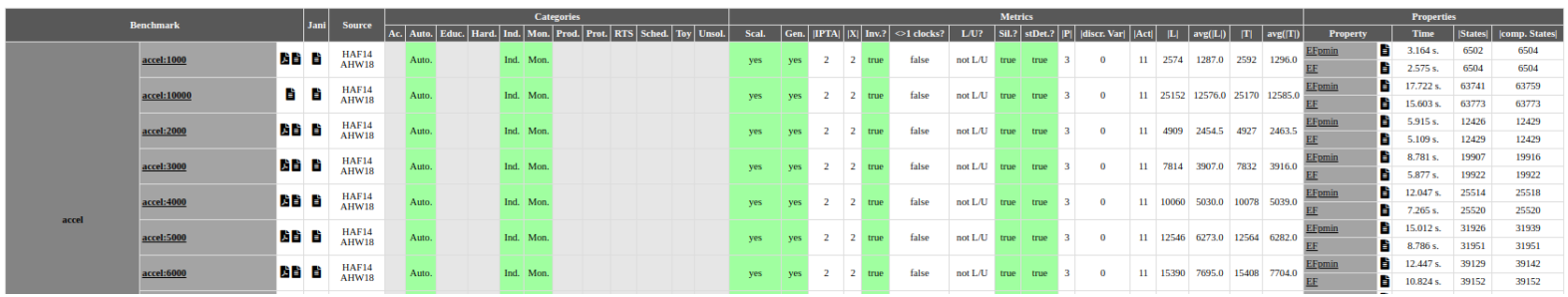
For example

Get all benchmarks

- that are scalable,
- that feature no invariant nor unobservable actions
- and for which the synthesis time is (approximately) between 10 and 60 seconds

Where to find it?

- Initial version DOI [AMP21a]
- A webpage (updated version) [AMP21b]



Benchmark		Jani	Source	Categories												Metrics														Properties			
				Ac.	Auto.	Educ.	Hard.	Ind.	Mon.	Prod.	Prot.	RTS	Sched.	Toy	Unsol.	Scal.	Gen.	|IPTA|	|X|	Inv.?	<>1 clocks?	L/U?	Sil.?	stDet.?	|P|	|discr. Var|	|Act|	|L|	avg(|L|)	|T|	avg(|T|)	Property	Time	|States|	|comp. States|
accel	accel:1000		HAF14 AHW18		Auto.			Ind.	Mon.							yes	yes	2	2	true	false	not L/U	true	true	3	0	11	2574	1287.0	2592	1296.0	EFomin	3.164 s.	6502	6504
																																EF	2.575 s.	6504	6504
	accel:10000		HAF14 AHW18		Auto.			Ind.	Mon.							yes	yes	2	2	true	false	not L/U	true	true	3	0	11	25152	12576.0	25170	12585.0	EFomin	17.722 s.	63741	63759
																																EF	15.603 s.	63773	63773
	accel:2000		HAF14 AHW18		Auto.			Ind.	Mon.							yes	yes	2	2	true	false	not L/U	true	true	3	0	11	4909	2454.5	4927	2463.5	EFomin	5.915 s.	12426	12429
																																EF	5.109 s.	12429	12429
	accel:3000		HAF14 AHW18		Auto.			Ind.	Mon.							yes	yes	2	2	true	false	not L/U	true	true	3	0	11	7814	3907.0	7832	3916.0	EFomin	8.781 s.	19907	19916
																																EF	5.877 s.	19922	19922
	accel:4000		HAF14 AHW18		Auto.			Ind.	Mon.							yes	yes	2	2	true	false	not L/U	true	true	3	0	11	10060	5030.0	10078	5039.0	EFomin	12.047 s.	25514	25518
																																EF	7.265 s.	25520	25520
	accel:5000		HAF14 AHW18		Auto.			Ind.	Mon.							yes	yes	2	2	true	false	not L/U	true	true	3	0	11	12546	6273.0	12564	6282.0	EFomin	15.012 s.	31926	31939
																																EF	8.786 s.	31951	31951
	accel:6000		HAF14 AHW18		Auto.			Ind.	Mon.							yes	yes	2	2	true	false	not L/U	true	true	3	0	11	15390	7695.0	15408	7704.0	EFomin	12.447 s.	39129	39142
																																EF	10.824 s.	39152	39152

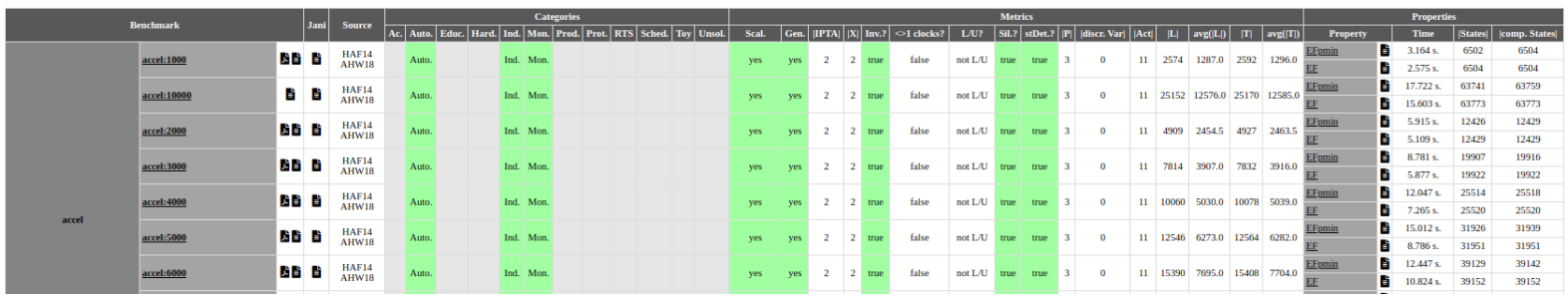
- with the static and semantic metrics
- with expected results (if exist)

[AMP21a] Étienne André, Dylan Marinho, and Jaco van de Pol. *The IMITATOR benchmarks library 2.0: A benchmarks library for extended parametric timed automata*. Apr. 2021. DOI: 10.5281/zenodo.4730980

[AMP21b] Étienne André, Dylan Marinho, and Jaco van de Pol. *The IMITATOR benchmarks library v2.0*. 2021. URL: https://www.imitator.fr/static/library2/

Where to find it?

- Initial version DOI [AMP21a]
- A webpage (updated version) [AMP21b]



Benchmark		Jani	Source	Categories												Metrics														Properties			
				Ac.	Auto.	Educ.	Hard.	Ind.	Mon.	Prod.	Prot.	RTS	Sched.	Toy	Unsol.	Scal.	Gen.	|IPTA|	|X|	Inv.?	<>1 clocks?	L/U?	Sil.?	stDet.?	|P|	|discr. Var|	|Act|	|L|	avg(|L|)	|T|	avg(|T|)	Property	Time	|States|	|comp. States|
accel	accel:1000		HAF14 AHW18		Auto.			Ind.	Mon.							yes	yes	2	2	true	false	not L/U	true	true	3	0	11	2574	1287.0	2592	1296.0	EFomin	3.164 s.	6502	6504
																																EF	2.575 s.	6504	6504
	accel:10000		HAF14 AHW18		Auto.			Ind.	Mon.							yes	yes	2	2	true	false	not L/U	true	true	3	0	11	25152	12576.0	25170	12585.0	EFomin	17.722 s.	63741	63759
																																EF	15.603 s.	63773	63773
	accel:2000		HAF14 AHW18		Auto.			Ind.	Mon.							yes	yes	2	2	true	false	not L/U	true	true	3	0	11	4909	2454.5	4927	2463.5	EFomin	5.915 s.	12426	12429
																																EF	5.109 s.	12429	12429
	accel:3000		HAF14 AHW18		Auto.			Ind.	Mon.							yes	yes	2	2	true	false	not L/U	true	true	3	0	11	7814	3907.0	7832	3916.0	EFomin	8.781 s.	19907	19916
																																EF	5.877 s.	19922	19922
	accel:4000		HAF14 AHW18		Auto.			Ind.	Mon.							yes	yes	2	2	true	false	not L/U	true	true	3	0	11	10060	5030.0	10078	5039.0	EFomin	12.047 s.	25514	25518
																																EF	7.265 s.	25520	25520
	accel:5000		HAF14 AHW18		Auto.			Ind.	Mon.							yes	yes	2	2	true	false	not L/U	true	true	3	0	11	12546	6273.0	12564	6282.0	EFomin	15.012 s.	31926	31939
																																EF	8.786 s.	31951	31951
	accel:6000		HAF14 AHW18		Auto.			Ind.	Mon.							yes	yes	2	2	true	false	not L/U	true	true	3	0	11	15390	7695.0	15408	7704.0	EFomin	12.447 s.	39129	39142
																																EF	10.824 s.	39152	39152

  - with the static and semantic metrics
  - with expected results (if exist)

We are open to new model submissions!

[AMP21a] Étienne André, Dylan Marinho, and Jaco van de Pol. *The IMITATOR benchmarks library 2.0: A benchmarks library for extended parametric timed automata*. Apr. 2021. DOI: 10.5281/zenodo.4730980

[AMP21b] Étienne André, Dylan Marinho, and Jaco van de Pol. *The IMITATOR benchmarks library v2.0*. 2021. URL: https://www.imitator.fr/static/library2/

Future work

- Keep enlarging the library using future benchmarks, and future synthesis algorithms
  - Add "difficulty" score to benchmarks (from easy to unsolvable)
- Bindings with other libraries, e. g.,
  - share a non-parametric version of PTAs to TAs libraries
  - integrate benchmarks from hybrid automata libraries to ours when the expressiveness is compatible
- Develop translations to other model checkers: e. g.,
  - `Uppaal` [LPY97] (non-parametric timed automata)
  - `SpaceEx` [Fre+11] (hybrid systems)
  - `Roméo` [Lim+09] (parametric time Petri nets)

[LPY97] Kim Guldstrand Larsen, Paul Pettersson, and Wang Yi. "UPPAAL in a Nutshell". In: *International Journal on Software Tools for Technology Transfer* 1.1-2 (1997), pp. 134–152

[Fre+11] Goran Frehse et al. "SpaceEx: Scalable Verification of Hybrid Systems". In: *CAV*. vol. 6806. Lecture Notes in Computer Science. Springer, 2011, pp. 379–395

[Lim+09] Didier Lime et al. "Romeo: A Parametric Model-Checker for Petri Nets with Stopwatches". In: *TACAS*. vol. 5505. Lecture Notes in Computer Science. Springer, Mar. 2009, pp. 54–57

References I

Rajeev Alur and David L. Dill. "A theory of timed automata". In: *Theoretical Computer Science* 126.2 (Apr. 1994), pp. 183–235. ISSN: 0304-3975. DOI: 10.1016/0304-3975(94)90010-8.

Rajeev Alur, Thomas A. Henzinger, and Moshe Y. Vardi. "Parametric real-time reasoning". In: *STOC* (May 16–18, 1993). Ed. by S. Rao Kosaraju, David S. Johnson, and Alok Aggarwal. San Diego, California, United States: ACM, 1993, pp. 592–601. ISBN: 0-89791-591-7. DOI: 10.1145/167088.167242.

References II


Étienne André, Ichiro Hasuo, and Masaki Waga. "Offline timed pattern matching under uncertainty". In: *ICECCS* (Dec. 12–14, 2018). Ed. by Anthony Widjaja Lin and Jun Sun. Melbourne, Australia: IEEE Computer Society, 2018, pp. 10–20. DOI: 10.1109/ICECCS2018.2018.00010.


Étienne André, Dylan Marinho, and Jaco van de Pol. *The IMITATOR benchmarks library 2.0: A benchmarks library for extended parametric timed automata*. Apr. 2021. DOI: 10.5281/zenodo.4730980.


Étienne André, Dylan Marinho, and Jaco van de Pol. *The IMITATOR benchmarks library v2.0*. 2021. URL: https://www.imitator.fr/static/library2/.

References III

Étienne André. *The IMITATOR benchmarks library v1.0*. 2018. URL: https://www.imitator.fr/library1.html.

Étienne André. "IMITATOR 3: Synthesis of timing parameters beyond decidability". In: *CAV* (July 18–23, 2021). Ed. by Rustan Leino and Alexandra Silva. virtual, 2021.

Patricia Bouyer, Nicolas Markey, and Ocan Sankur. "Robustness in timed automata". In: *RP*. Ed. by Parosh Aziz Abdulla and Igor Potapov. Vol. 8169. Lecture Notes in Computer Science. Invited paper. Uppsala, Sweden: Springer, Sept. 2013, pp. 1–18. DOI: 10.1007/978-3-642-41036-9_1.

References IV

Carlos E. Budde et al. *JANI specification*. 2017. URL: https://jani-spec.org/.

Goran Frehse et al. "SpaceEx: Scalable Verification of Hybrid Systems". In: *CAV*. Ed. by Ganesh Gopalakrishnan and Shaz Qadeer. Vol. 6806. Lecture Notes in Computer Science. Snowbird, UT, USA: Springer, 2011, pp. 379–395. DOI: 10.1007/978-3-642-22110-1_30.

References V

Didier Lime et al. "Romeo: A Parametric Model-Checker for Petri Nets with Stopwatches". In: *TACAS* (Mar. 22–29, 2009). Ed. by Stefan Kowalewski and Anna Philippou. Vol. 5505. Lecture Notes in Computer Science. York, United Kingdom: Springer, Mar. 2009, pp. 54–57. DOI: 10.1007/978-3-642-00768-2_6.

Kim Guldstrand Larsen, Paul Pettersson, and Wang Yi. "UPPAAL in a Nutshell". In: *International Journal on Software Tools for Technology Transfer* 1.1-2 (1997), pp. 134–152. DOI: 10.1007/s100090050010.

References VI

Lars Luthmann et al. "Modeling and Testing Product Lines with Unbounded Parametric Real-Time Constraints". In: *SPLC, Volume A* (Sept. 25–29, 2017). Ed. by Myra B. Cohen et al. Sevilla, Spain: ACM, 2017, pp. 104–113. DOI: 10.1145/3106195.3106204.