SynCoP

April 22, 2023
Paris, France

Execution-time opacity problems in (parametric) timed automata

Dylan Marinho

Université de Lorraine, CNRS, Inria, LORIA, Nancy, France

Join works with Étienne André, Engel Lefaucheux, Didier Lime, and Sun Jun

These works are partially supported by the ANR-NRF research program ProMiS (ANR-19-CE25-0015)
and the ANR research program BisoUS (ANR-22-CE48-0012).

Context: timing attacks

- Principle: deduce private information from timing data (execution time)

Issues:

- May depend on the implementation (or, even worse, be introduced by the compiler)
- A relatively trivial solution: make the program last always its maximum execution time
  Drawback: loss of efficiency

$\rightsquigarrow$ Non-trivial problem

A simple example of timing attack

```
# input pwd     : Real password
# input attempt: Tentative password
for i = 0 to min(len(pwd), len(attempt)) - 1 do
    if pwd[i] =/= attempt[i] then
        return false
done
return true
```

A simple example of timing attack

```
# input pwd     : Real password
# input attempt: Tentative password
for i = 0 to min(len(pwd), len(attempt)) − 1 do
    if pwd[i] =/= attempt[i] then
        return false
done
return true
```

pwd	c	h	i	c	k	e	n
attempt	c	h	e	e	s	e	

Execution time:

A simple example of timing attack

```
# input pwd     : Real password
# input attempt: Tentative password
for i = 0 to min(len(pwd), len(attempt)) - 1 do
    if pwd[i] =/= attempt[i] then
        return false
done
return true
```

pwd	c	h	i	c	k	e	n
attempt	c	h	e	e	s	e	

Execution time: ϵ

A simple example of timing attack

```
# input pwd     : Real password
# input attempt: Tentative password
for i = 0 to min(len(pwd), len(attempt)) - 1 do
    if pwd[i] =/= attempt[i] then
        return false
done
return true
```

pwd	c	h	i	c	k	e	n
attempt	c	h	e	e	s	e	

Execution time: $\epsilon + \epsilon$

A simple example of timing attack

```
# input pwd     : Real password
# input attempt: Tentative password
for i = 0 to min(len(pwd), len(attempt)) − 1 do
    if pwd[i] =/= attempt[i] then
        return false
done
return true
```

pwd	c	h	i	c	k	e	n
attempt	c	h	e	e	s	e	

Execution time: $\epsilon + \epsilon + \epsilon$

A simple example of timing attack

```
# input pwd     : Real password
# input attempt: Tentative password
for i = 0 to min(len(pwd), len(attempt)) − 1 do
    if pwd[i] =/= attempt[i] then
        return false
done
return true
```

pwd	c	h	i	c	k	e	n
attempt	c	h	e	e	s	e	

Execution time: $\epsilon + \epsilon + \epsilon$

- Problem: The execution time is proportional to the number of consecutive correct characters from the beginning of `attempt`

Informal problems

Question: can we exhibit secure execution times?

Computation problem: Execution-time opacity computation

Exhibit execution times for which it is not possible to infer information on the internal behavior

Informal problems

Question: can we exhibit secure execution times?

Computation problem: Execution-time opacity computation

Exhibit execution times for which it is not possible to infer information on the internal behavior

Question: can we make sure all execution times are secure?

Decision problem: Full execution-time opacity

Can we decide whether it is impossible to infer information on the internal behavior, whatever (for all) execution times?

Informal parametric problems

Further question: can we also tune internal timing constants to make the system resisting to timing attacks?

Synthesis problem: Execution-time opacity synthesis

Exhibit execution times and internal timing constants for which it is not possible to infer information on the internal behavior

Outline

Outline

ET-opacity problems in timed automata

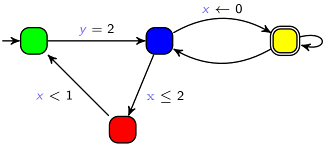
- Timed model checking and timed automata
- Execution-Time Opacity Problems
- Expiring-ET-Opacity Problems

ET-opacity parametrization

Results

Perspectives

Timed model checking

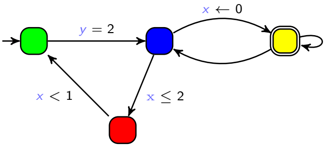


A model of the system

 is unreachable

A property to be satisfied

Timed model checking



A model of the system

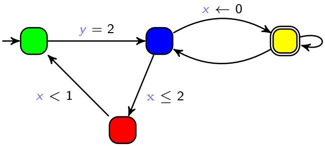
$\models$?

 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

$\overset{?}{\models}$

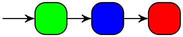
 is unreachable

A property to be satisfied

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

Yes

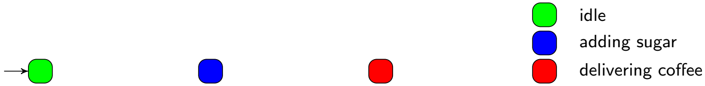
No



Counterexample

Timed automaton (TA)

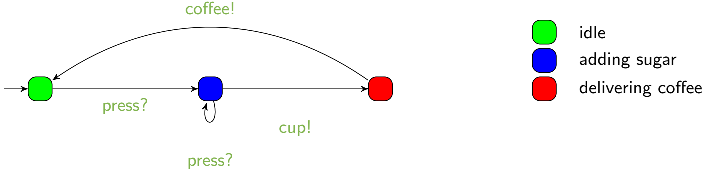
- Finite state automaton (sets of locations)



[AD94] Rajeev Alur and David L. Dill. "A theory of timed automata". In: *Theoretical Computer Science* 126.2 (Apr. 1994), pp. 183–235. DOI: 10.1016/0304-3975(94)90010-8

Timed automaton (TA)

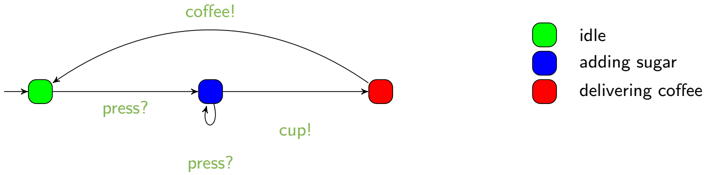
- Finite state automaton (sets of locations and actions)



[AD94] Rajeev Alur and David L. Dill. “A theory of timed automata”. In: *Theoretical Computer Science* 126.2 (Apr. 1994), pp. 183–235. DOI: 10.1016/0304-3975(94)90010-8

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

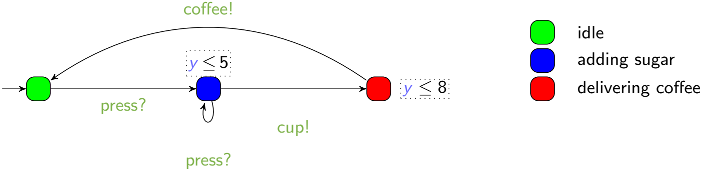


[AD94] Rajeev Alur and David L. Dill. "A theory of timed automata". In: *Theoretical Computer Science* 126.2 (Apr. 1994), pp. 183–235. DOI: 10.1016/0304-3975(94)90010-8

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

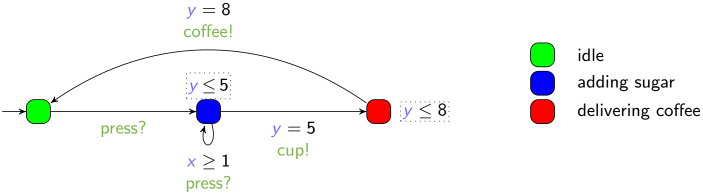


[AD94] Rajeev Alur and David L. Dill. "A theory of timed automata". In: *Theoretical Computer Science* 126.2 (Apr. 1994), pp. 183–235. DOI: 10.1016/0304-3975(94)90010-8

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

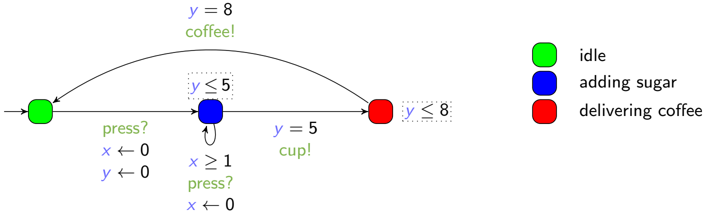


[AD94] Rajeev Alur and David L. Dill. "A theory of timed automata". In: *Theoretical Computer Science* 126.2 (Apr. 1994), pp. 183–235. DOI: 10.1016/0304-3975(94)90010-8

Timed automaton (TA)

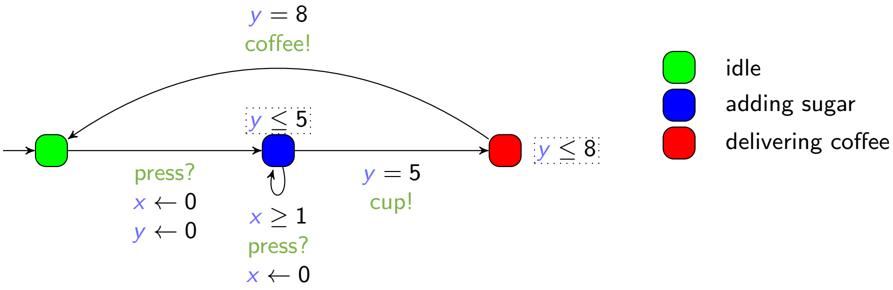
- Finite state automaton (sets of locations and actions) augmented with a set X of clocks [AD94]
  - Real-valued variables evolving linearly at the same rate
  - Can be compared to integer constants in invariants and guards

- 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

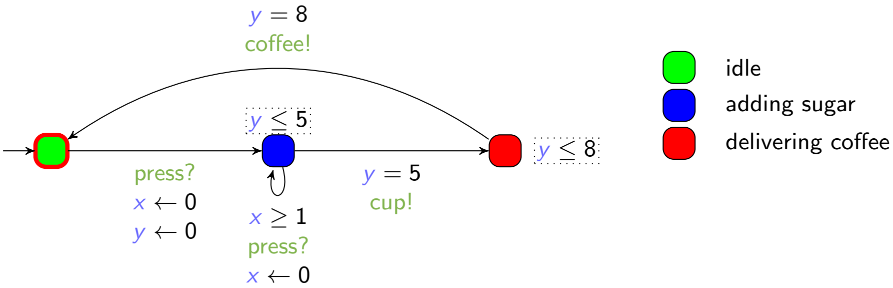


[AD94] Rajeev Alur and David L. Dill. "A theory of timed automata". In: *Theoretical Computer Science* 126.2 (Apr. 1994), pp. 183–235. DOI: 10.1016/0304-3975(94)90010-8

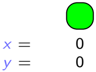
The most critical system: The coffee machine



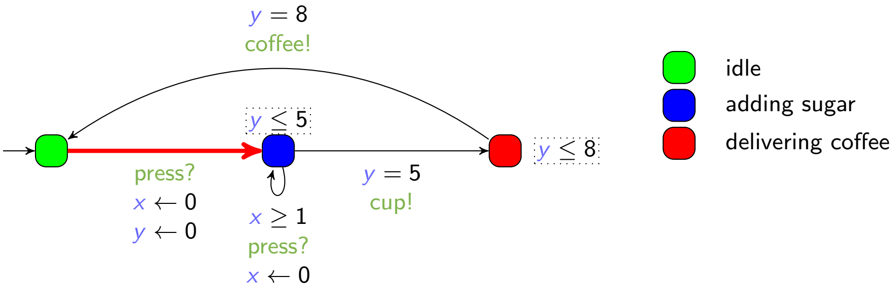
The most critical system: The coffee machine



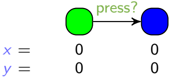
- Example of concrete run for the coffee machine
  - Coffee with 2 doses of sugar



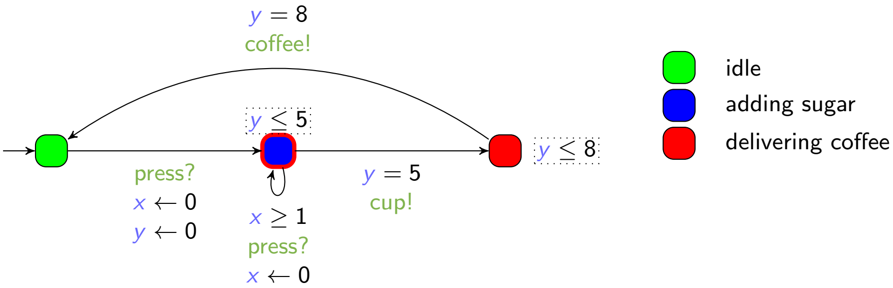
The most critical system: The coffee machine



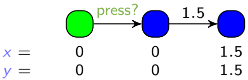
- Example of concrete run for the coffee machine
  - Coffee with 2 doses of sugar



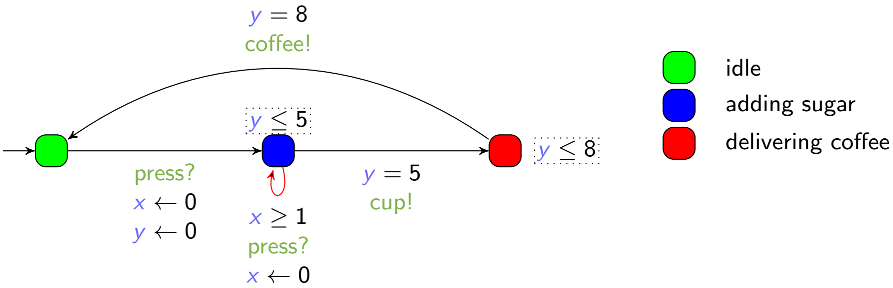
The most critical system: The coffee machine



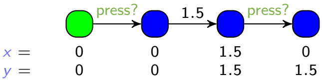
- Example of concrete run for the coffee machine
  - Coffee with 2 doses of sugar



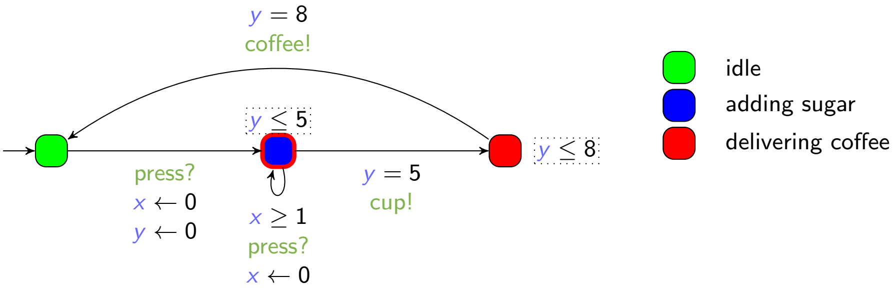
The most critical system: The coffee machine



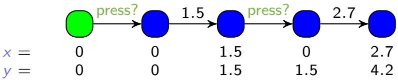
- Example of concrete run for the coffee machine
  - Coffee with 2 doses of sugar



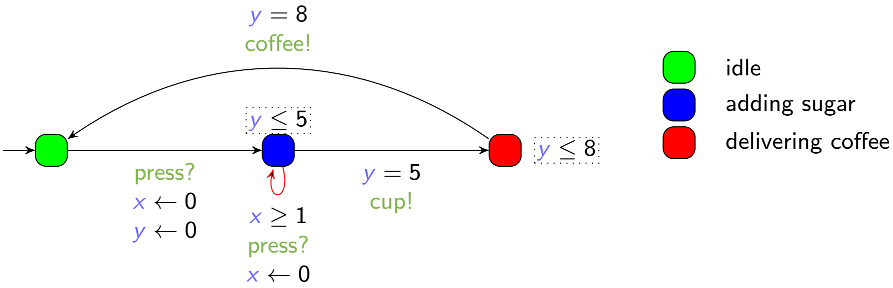
The most critical system: The coffee machine



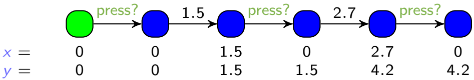
- Example of concrete run for the coffee machine
  - Coffee with 2 doses of sugar



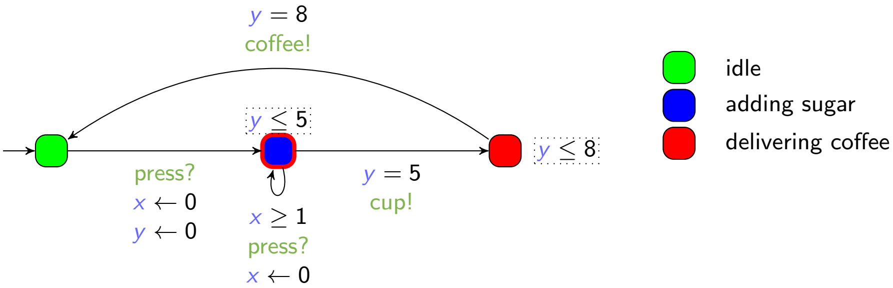
The most critical system: The coffee machine



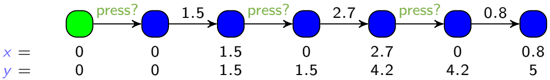
- Example of concrete run for the coffee machine
  - Coffee with 2 doses of sugar



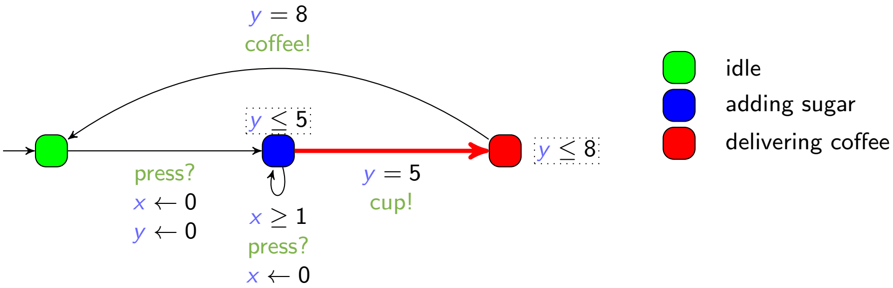
The most critical system: The coffee machine



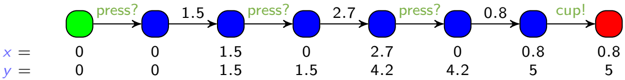
- Example of concrete run for the coffee machine
  - Coffee with 2 doses of sugar



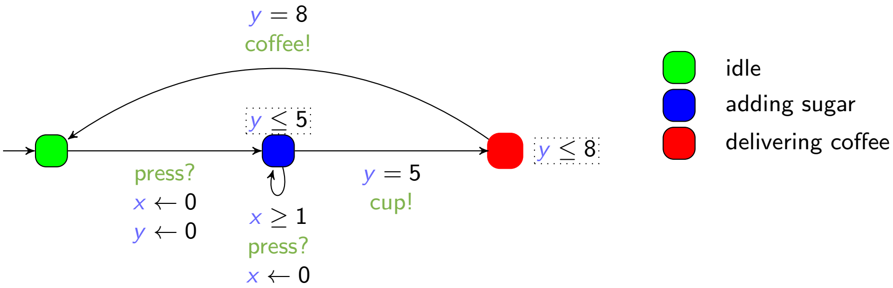
The most critical system: The coffee machine



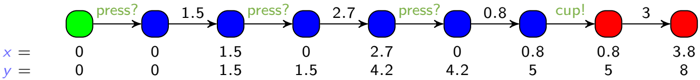
- Example of concrete run for the coffee machine
  - Coffee with 2 doses of sugar



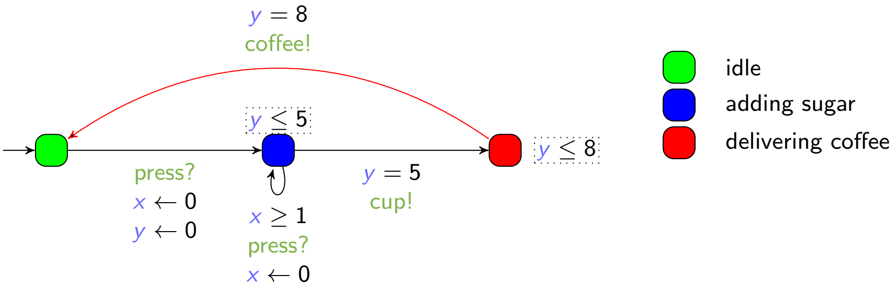
The most critical system: The coffee machine



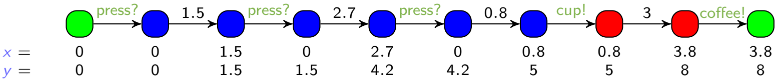
- Example of concrete run for the coffee machine
  - Coffee with 2 doses of sugar



The most critical system: The coffee machine



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

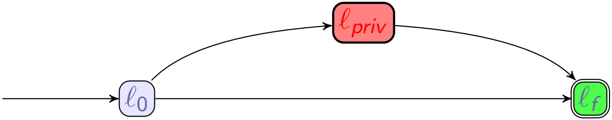


Outline

Formalization

Hypotheses: [AS19]

- A start location ℓ_0 and an end location ℓ_f
- A special private location ℓ_{priv}



Definition (execution-time opacity)

The system is ET-opaque for a duration d if there exist two runs to ℓ_f of duration d

1. one visiting ℓ_{priv}
2. one *not* visiting ℓ_{priv}

[AS19] Étienne André and Jun Sun. "Parametric Timed Model Checking for Guaranteeing Timed Opacity". In: *ATVA* (Oct. 28–31, 2019). Ed. by Yu-Fang Chen, Chih-Hong Cheng, and Javier Esparza. Vol. 11781. Lecture Notes in Computer Science. Taipei, Taiwan: Springer, 2019, pp. 115–130. DOI: 10.1007/978-3-030-31784-3_7

Three levels of ET-opacity

Existential – ∃

There exist two runs of duration d,
one visiting ℓ_{priv},
one not visiting ℓ_{priv}

Weak

For all duration d,
There exists a run of duration d visiting ℓ_{priv}
$\Rightarrow$
There exists a run of duration d not visiting ℓ_{priv}

Full

For all duration d,
There exists a run of duration d visiting ℓ_{priv}
$\Leftrightarrow$
There exists a run of duration d not visiting ℓ_{priv}

Three levels of ET-opacity

Existential – $\exists$

private durations $\cap$ public durations $\neq \emptyset$

Weak

For all duration d,
There exists a run of duration d visiting ℓ_{priv}
$\Rightarrow$
There exists a run of duration d not visiting ℓ_{priv}

Full

For all duration d,
There exists a run of duration d visiting ℓ_{priv}
$\Leftrightarrow$
There exists a run of duration d not visiting ℓ_{priv}

Three levels of ET-opacity

Existential – $\exists$

private durations $\cap$ public durations $\neq \emptyset$

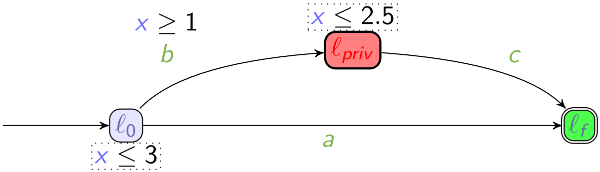
Weak

private durations $\subseteq$ public durations

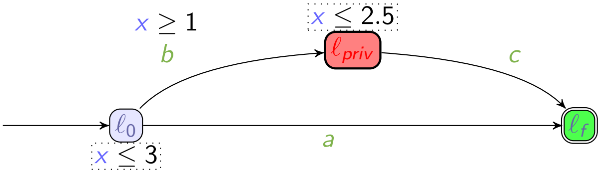
Full

private durations $=$ public durations

Example

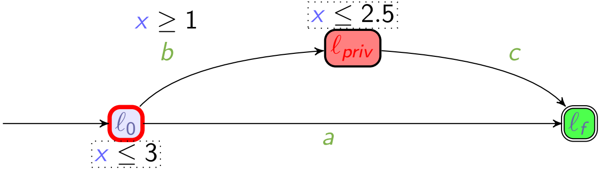


Example



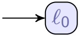
- There exist *(at least)* two runs of duration $d = 2$:

Example

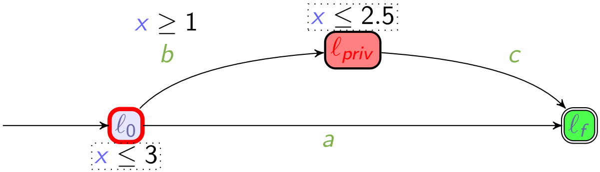


- There exist *(at least)* two runs of duration $d = 2$:

visiting ℓ_{priv}

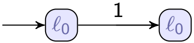


Example

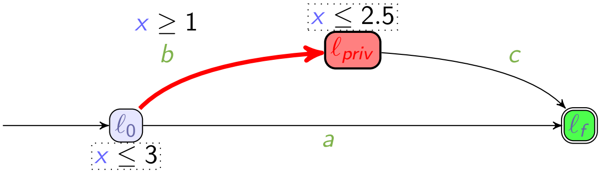


- There exist *(at least)* two runs of duration $d = 2$:

visiting ℓ_{priv}

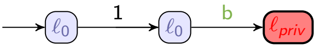


Example

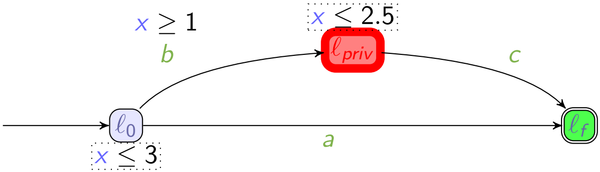


- There exist *(at least)* two runs of duration $d = 2$:

visiting ℓ_{priv}

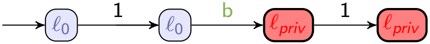


Example

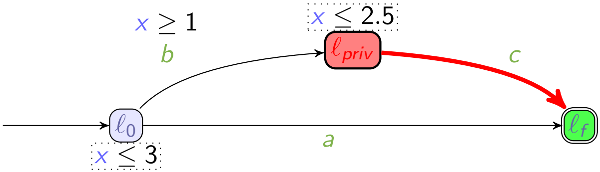


- There exist *(at least)* two runs of duration $d = 2$:

visiting ℓ_{priv}

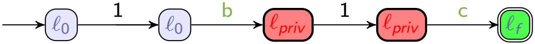


Example

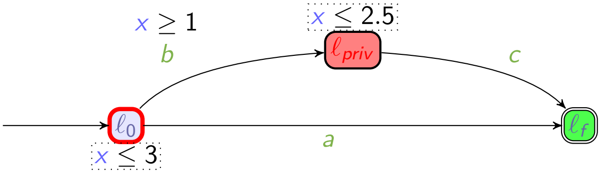


- There exist *(at least)* two runs of duration $d = 2$:

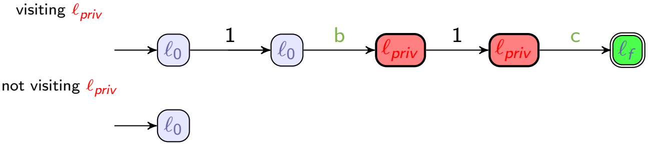
visiting ℓ_{priv}



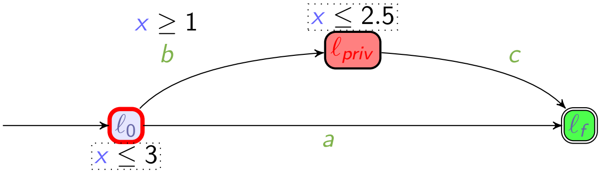
Example



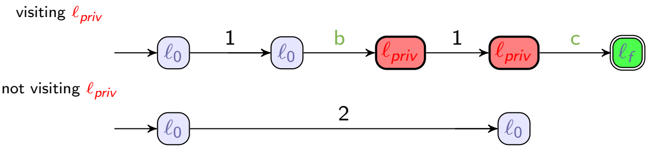
- There exist *(at least)* two runs of duration $d = 2$:



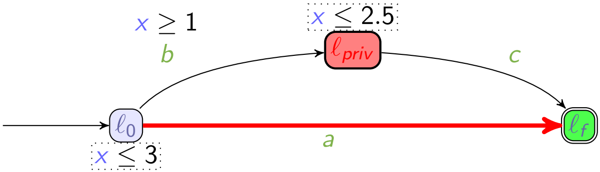
Example



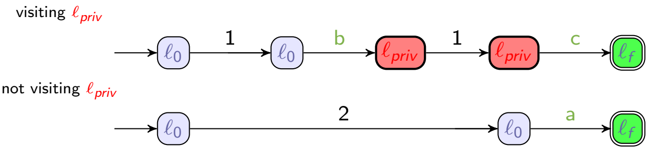
- There exist *(at least)* two runs of duration $d = 2$:



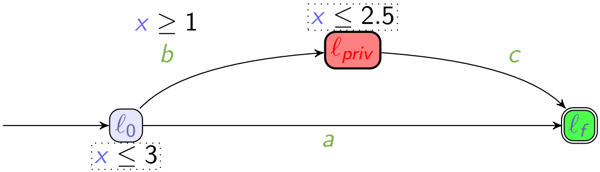
Example



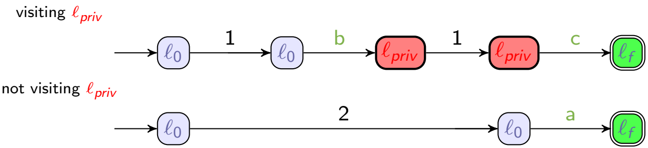
- There exist *(at least)* two runs of duration $d = 2$:



Example



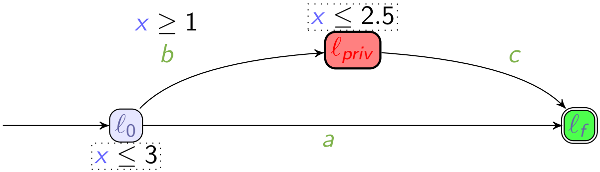
- There exist *(at least)* two runs of duration $d = 2$:



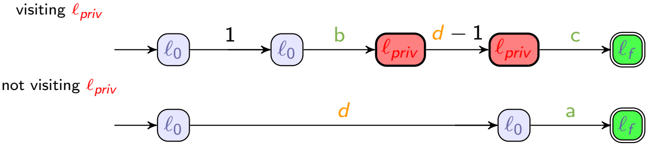
The system is ET-opaque for a duration $d = 2$

The system is $\exists$-ET-opaque

Example



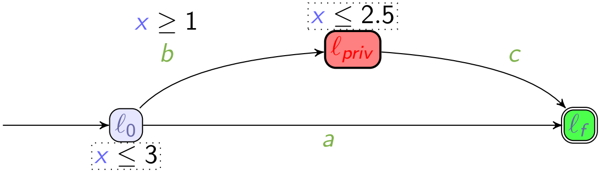
- There exist *(at least)* two runs of duration d for all durations $d \in [1, 2.5]$:



The system is ET-opaque for all durations in $[1, 2.5]$

The system is $\exists$-ET-opaque

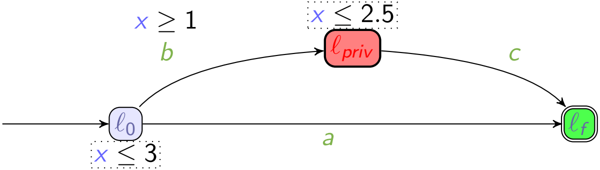
Example



- There exist *(at least)* two runs of duration d for all durations $d \in [1, 2.5]$

The system is ∃-ET-opaque

Example

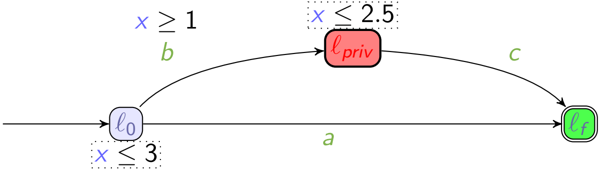


- There exist *(at least)* two runs of duration d for all durations $d \in [1, 2.5]$

The system is $\exists$-ET-opaque

- But,
  - private execution times are $[1, 2.5]$
    public execution times are $[0, 3]$

Example

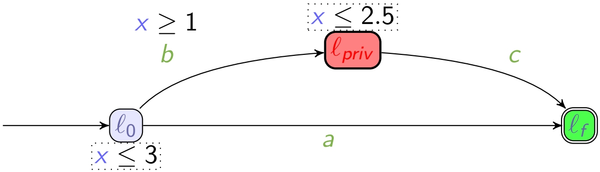


- There exist *(at least)* two runs of duration d for all durations $d \in [1, 2.5]$

The system is ∃-ET-opaque

- But,
  - private execution times are $[1, 2.5]$
    public execution times are $[0, 3]$
  - private durations $\subseteq$ public durations

Example



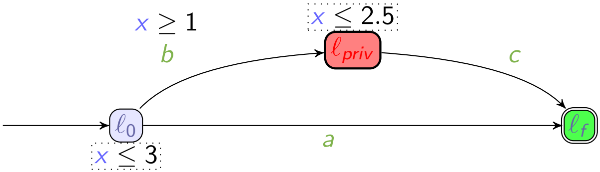
- There exist *(at least)* two runs of duration d for all durations $d \in [1, 2.5]$

The system is ∃-ET-opaque

- But,
  - private execution times are $[1, 2.5]$
    public execution times are $[0, 3]$
  - private durations $\subseteq$ public durations

The system is weakly ET-opaque

Example



- There exist *(at least)* two runs of duration d for all durations $d \in [1, 2.5]$

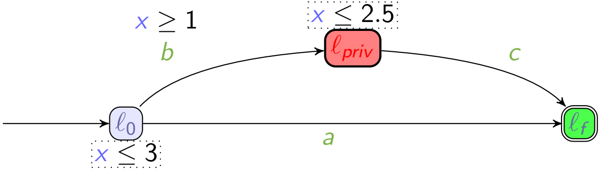
The system is $\exists$-ET-opaque

- But,
  - private execution times are $[1, 2.5]$
    public execution times are $[0, 3]$
  - private durations $\subseteq$ public durations

The system is weakly ET-opaque

  - private durations $\neq$ public durations

Example



- There exist *(at least)* two runs of duration d for all durations $d \in [1, 2.5]$

The system is ∃-ET-opaque

- But,
  - private execution times are $[1, 2.5]$
    public execution times are $[0, 3]$
  - private durations $\subseteq$ public durations

The system is weakly ET-opaque

  - private durations $\neq$ public durations

The system is *not* fully ET-opaque

Outline

Expiring ET-opacity

Idea

The secret can expire: beyond a certain duration, knowing the secret is useless to the attacker (e. g., a cache value)

	Secret runs	**Non-secret runs**
ET-opacity	Runs visiting the private location (= private runs)	Runs not visiting the private location (= public runs)
expiring-ET-opacity	Private runs with ℓ_{priv} visit $\leq \Delta$ before the system completion	(i) Public runs and (ii) Private runs with ℓ_{priv} visit $> \Delta$ before the system completion

Three levels of ET-opacity

Existential–∃

private durations $\cap$ public durations $\neq \emptyset$

Weak

private durations $\subseteq$ public durations

Full

private durations $=$ public durations

Three levels of expiring ET-opacity

Existential–∃ expiring

secret durations $\cap$ non-secret durations $\neq \emptyset$

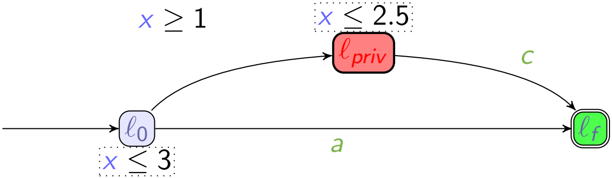
Weak expiring

secret durations $\subseteq$ non-secret durations

Full expiring

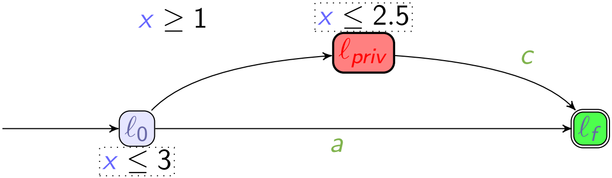
secret durations $=$ non-secret durations

Example



ET-opacity notion		Secret	Non secret	Answer
	$\exists$			$\surd$
	weak	$[1, 2.5]$	$[0, 3]$	$\surd$
	full			$\times$
	$\exists$-exp.			$\surd$
$\Delta = 1$	weak-exp.	$[1, 2.5]$	$(2, 2.5] \cup [0, 3]$	$\surd$
	full-exp.			$\times$

Example



ET-opacity notion		Secret	Non secret	Answer
	$\exists$			$\surd$
	weak	$[1, 2.5]$	$[0, 3]$	$\surd$
	full			$\times$
	$\exists$-exp.			$\surd$
$\Delta = 1$	weak-exp.	$[1, 2.5]$	$(2, 2.5] \cup [0, 3]$	$\surd$
	full-exp.			$\times$
	$\exists$-exp.			$\surd$
$\Delta = 1.25$	weak-exp.	$[1, 2.5]$	$(2.25, 2.5] \cup [0, 3]$	$\surd$
	full-exp.			$\times$

Outline

ET-opacity problems in timed automata

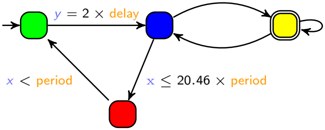
ET-opacity parametrization

Results

Perspectives

Outline

timed model checking



A model of the system

$\models$?

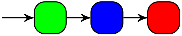
 is unreachable

A property to be satisfied

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

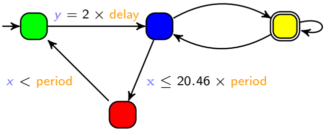
Yes

No



Counterexample

Parametric timed model checking



A model of the system

$\models$?

 is unreachable

A property to be satisfied

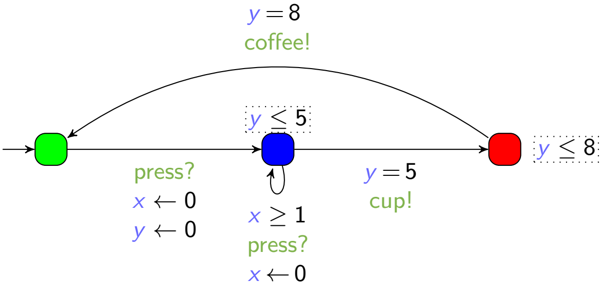
- Question: for what values of the parameters does the model of the system satisfy the property?

Yes if. . .

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

Timed Automaton (PTA)

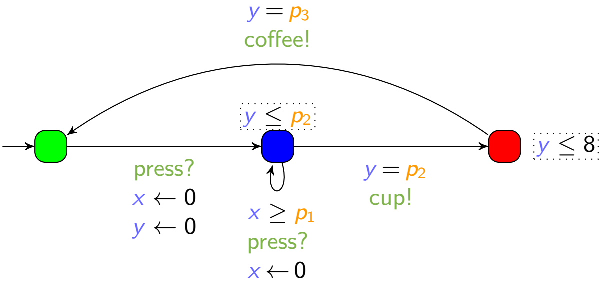
- Timed automaton (sets of locations, actions and clocks)



[AHV93] Rajeev Alur, Thomas A. Henzinger, and Moshe Y. Vardi. "Parametric real-time reasoning". In: *STOC* (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. DOI: 10.1145/167088.167242

Parametric Timed Automaton (PTA)

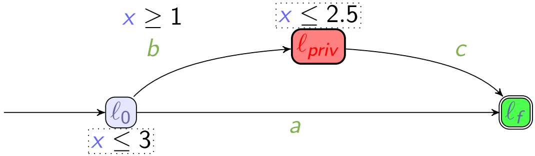
- Timed automaton (sets of locations, actions and clocks) augmented with a set P of parameters [AHV93]
  - Unknown constants compared to a clock in guards and invariants



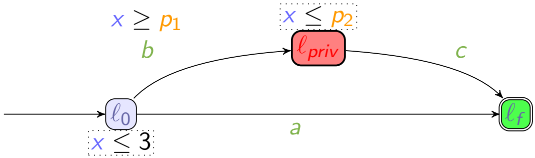
[AHV93] Rajeev Alur, Thomas A. Henzinger, and Moshe Y. Vardi. "Parametric real-time reasoning". In: *STOC* (May 16–18, 1993). Ed. by S. Rao Kosaraju, David S. Johnson, and Alok Aggarwal. San Diego, California, United States: ACM, 1993, pp. 592–601. DOI: 10.1145/167088.167242

Outline

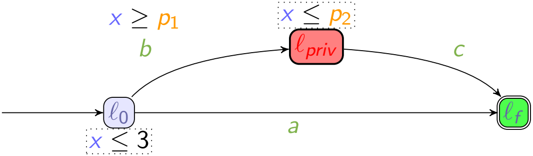
Example



Example

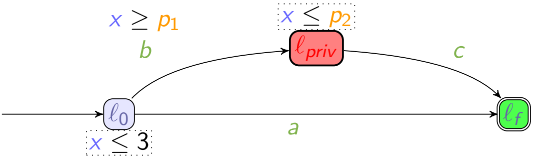


Example



Private	$[p_1, p_2]$
Public	$[0, 3]$

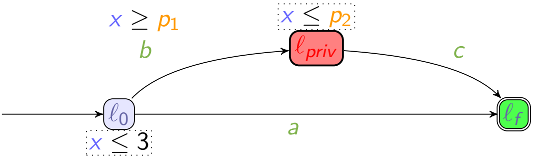
Example



Private	$[p_1, p_2]$
Public	$[0, 3]$

ET-opacity notion	Private	Public	Answer
$p_1 = 1 \wedge p_2 = 2.5$			
$\exists$			$\surd$
weak	$[1, 2.5]$	$[0, 3]$	$\surd$
full			$\times$

Example



Private	$[p_1, p_2]$
Public	$[0, 3]$

ET-opacity notion	**Private**	**Public**	**Answer**
$p_1 = 1 \wedge p_2 = 2.5$			
$\exists$			$\surd$
weak	$[1, 2.5]$	$[0, 3]$	$\surd$
full			$\times$
$p_1 = 0 \wedge p_2 = 3$			
$\exists$			$\surd$
weak	$[0, 3]$	$[0, 3]$	$\surd$
full			$\surd$

Two classes of parametric problems

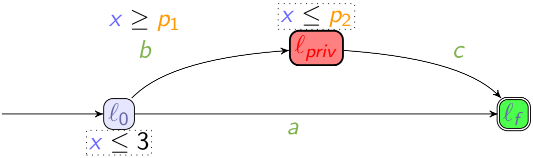
p-Emptiness problem

Is the set of parameter valuations ensuring the property empty?

p-Synthesis problem

Synthesize all the parameter valuations ensuring the property

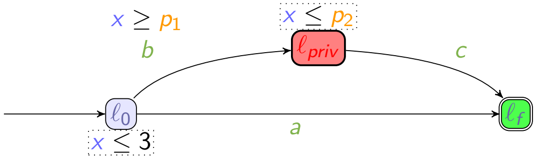
Example



Private	$[p_1, p_2]$
Public	$[0, 3]$

ET-opacity notion	p-Emptiness	p-Synthesis
$\exists$		
weak		
full		

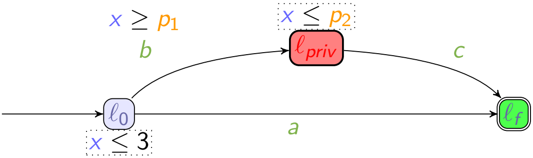
Example



Private	$[p_1, p_2]$
Public	$[0, 3]$

ET-opacity notion	p-Emptiness	p-Synthesis
$\exists$	$\times$ ($\exists v$)	
weak	$\times$ ($\exists v$)	
full	$\times$ ($\exists v$)	

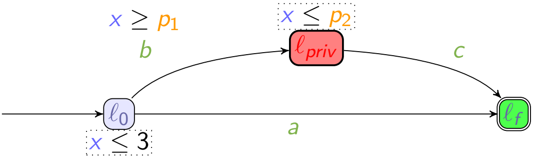
Example



Private	$[p_1, p_2]$
Public	$[0, 3]$

ET-opacity notion	p-Emptiness	p-Synthesis
$\exists$	$\times$ ($\exists v$)	$0 \leq p_1 \leq 3 \quad \wedge \quad p_1 \leq p_2$
weak	$\times$ ($\exists v$)	
full	$\times$ ($\exists v$)	

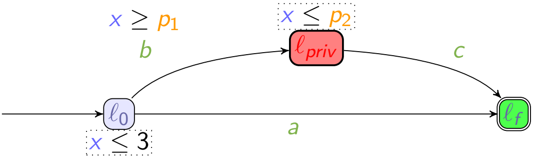
Example



Private	$[p_1, p_2]$
Public	$[0, 3]$

ET-opacity notion	p-Emptiness	p-Synthesis
$\exists$	$\times (\exists v)$	$0 \leq p_1 \leq 3 \quad \wedge \quad p_1 \leq p_2$
weak	$\times (\exists v)$	$0 \leq p_1 \wedge p_2 \leq 3 \quad \wedge \quad p_1 \leq p_2$
full	$\times (\exists v)$	

Example



Private	$[p_1, p_2]$
Public	$[0, 3]$

ET-opacity notion	p-Emptiness	p-Synthesis		
$\exists$	$\times_{(\exists v)}$	$0 \leq p_1 \leq 3$	$\wedge$	$p_1 \leq p_2$
weak	$\times_{(\exists v)}$	$0 \leq p_1 \wedge p_2 \leq 3$	$\wedge$	$p_1 \leq p_2$
full	$\times_{(\exists v)}$	$p_1 = 0 \wedge p_2 = 3$		

Outline

Outline

Summary of the results for ET-opacity [And+22]

		∃-ET-opaque	weakly ET-opaque	fully ET-opaque
Decision	TA	√	?	√
p-emptiness	L/U-PTA	√	?	×
	PTA	×	?	×
p-synthesis	L/U-PTA	×	?	×
	PTA	×	?	×

L/U-PTA (*Lower/Upper-PTA*): subclass of PTA where the parameters are partitioned into two sets (either compared to clocks as upperbound, or as lower bound) [BL09]

[BL09] Laura Bozzelli and Salvatore La Torre. "Decision problems for lower/upper bound parametric timed automata". In: *Formal Methods in System Design* 35.2 (2009), pp. 121–151. DOI: 10.1007/s10703-009-0074-0

[And+22] Étienne André, Didier Lime, Dylan Marinho, and Jun Sun. "Guaranteeing timed opacity using parametric timed model checking". In: *ACM Transactions on Software Engineering and Methodology* 31.4 (Oct. 2022), pp. 1–36. DOI: 10.1145/3502851

Outline

Summary of the results for expiring-ET-opacity [ALM23]

		$\exists$-expiring-ET-opaque	weakly expiring-ET-opaque	fully expiring-ET-opaque
Δ-emptiness	TA	?	$\surd$	$\surd$
Δ-synthesis	TA	?	$\surd$	?
$(p + \Delta)$-emptiness	L/U-PTA	?	$\times$	$\times$
	PTA	?	$\times$	$\times$
$(p + \Delta)$-synthesis	L/U-PTA	?	$\times$	$\times$
	PTA	?	$\times$	$\times$

[ALM23] Étienne André, Engel Lefaucheux, and Dylan Marinho. "Expiring opacity problems in parametric timed automata". In: *ICECCS* (June 12–16, 2023). Ed. by Yamine Ait-Ameur and Ferhat Khendek. Accepted. Toulouse, France, 2023

Outline

ET-opacity problems in timed automata

ET-opacity parametrization

Results

Perspectives

Perspectives

Theory

- Some restricted problems remain open

  e. g., PTA with one clock

- Study more restrictive sub-classes, with the hope to exhibit a decidable one

  Promising subclass: U-PTAs (only upper-bound parameters)

Perspectives

Theory

- Some restricted problems remain open

  e. g., PTA with one clock
- Study more restrictive sub-classes, with the hope to exhibit a decidable one

  Promising subclass: U-PTAs (only upper-bound parameters)

Algorithmic and implementation

- Automatic translation of programs to timed automata
- Repairing a non ET-opaque system

References I

[AD94] Rajeev Alur and David L. Dill. "A theory of timed automata". In: *Theoretical Computer Science* 126.2 (Apr. 1994), pp. 183–235. DOI: 10.1016/0304-3975(94)90010-8.

[AHV93] 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. DOI: 10.1145/167088.167242.

[ALM23] Étienne André, Engel Lefaucheux, and Dylan Marinho. "Expiring opacity problems in parametric timed automata". In: *ICECCS* (June 12–16, 2023). Ed. by Yamine Ait-Ameur and Ferhat Khendek. Accepted. Toulouse, France, 2023.

References II

[And+22] Étienne André, Didier Lime, Dylan Marinho, and Jun Sun. “Guaranteeing timed opacity using parametric timed model checking”. In: *ACM Transactions on Software Engineering and Methodology* 31.4 (Oct. 2022), pp. 1–36. DOI: 10.1145/3502851.

[AS19] Étienne André and Jun Sun. “Parametric Timed Model Checking for Guaranteeing Timed Opacity”. In: *ATVA* (Oct. 28–31, 2019). Ed. by Yu-Fang Chen, Chih-Hong Cheng, and Javier Esparza. Vol. 11781. Lecture Notes in Computer Science. Taipei, Taiwan: Springer, 2019, pp. 115–130. DOI: 10.1007/978-3-030-31784-3_7.

References III

[BL09] Laura Bozzelli and Salvatore La Torre. "Decision problems for lower/upper bound parametric timed automata". In: *Formal Methods in System Design* 35.2 (2009), pp. 121–151. DOI: 10.1007/s10703-009-0074-0.