JFR 2019

2019年10月10日
日本、東京

Monitoring efficace de propriétés quantitatives en temps-réel

Étienne André[1,2,3], Ichiro Hasuo[2,4] and Masaki Waga[2,4]

[1] Université de Lorraine, CNRS, Inria, LORIA, Nancy, France France
[2] National Institute of Informatics, Japan
[3] JFLI, UMI CNRS, Tokyo, Japan
[4] SOKENDAI (The Graduate University for Advanced Studies)

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

Motivation: automotive industry

- Modern cars embed several processors and produce logs

Motivation: automotive industry

- Modern cars embed several processors and produce logs

- Log: sequences of events and timestamps

```
start    2.3
gear1    5.8
gear2    9.2
gear3   18.5
gear2   42.1
```

Motivation: automotive industry

- Modern cars embed several processors and produce logs

  - Log: sequences of events and timestamps

```
start    2.3
gear1    5.8
gear2    9.2
gear3   18.5
gear2   42.1
```

- How to ensure on-the-fly that some properties are satisfied on a log?
  - "It never happens that gear1 and gear3 are separated by less than 5 s"

Motivation: automotive industry

- Modern cars embed several processors and produce logs

  - Log: sequences of events and timestamps

```
start    2.3
gear1    5.8
gear2    9.2
gear3   18.5
gear2   42.1
```

- How to ensure on-the-fly that some properties are satisfied on a log?
  - "It never happens that gear1 and gear3 are separated by less than 5 s"

$\Rightarrow$ Monitoring

Larger motivation: data collection and management

- Personal mobile devices collect large amounts of data

Larger motivation: data collection and management

- Personal mobile devices collect large amounts of data

These data can also come in the form of a timed log

```
start walking                                    2.3
```

Larger motivation: data collection and management

- Personal mobile devices collect large amounts of data

These data can also come in the form of a timed log

```
start walking                    2.3
walk faster                      6.3
```

Larger motivation: data collection and management

- Personal mobile devices collect large amounts of data

These data can also come in the form of a timed log

`start walking`	2.3
`walk faster`	6.3
`receive SMS`	15.8

Larger motivation: data collection and management

- Personal mobile devices collect large amounts of data

These data can also come in the form of a timed log

`start walking`	2.3
`walk faster`	6.3
`receive SMS`	15.8
`read SMS`	19.2

Larger motivation: data collection and management

- Personal mobile devices collect large amounts of data

These data can also come in the form of a timed log

```
start walking                                  2.3
walk faster                                    6.3
receive SMS                                   15.8
read SMS                                      19.2
sound of someone bumping into a lamp          22.5
```

Larger motivation: data collection and management

- Personal mobile devices collect large amounts of data

These data can also come in the form of a timed log

```
start walking                              2.3
walk faster                                6.3
receive SMS                               15.8
read SMS                                  19.2
sound of someone bumping into a lamp      22.5
```

- Key challenge: manage these data
  - Verify properties: “has the owner bumped into a street lamp”?
    - key applications (health, …)
  - Deduce information:
    - “what are the minimum/maximum intervals without visiting this shop”?
    - “is the user visiting this place more or less periodically?” (without knowing the actual period)

Outline

Untimed pattern matching: example

- Naive algorithm for pattern matching

$$\texttt{c}\quad\texttt{r}\quad\texttt{e}\quad\texttt{p}\quad\texttt{e}\quad\texttt{s}\quad \in ?L\left(\{c|i|d\}^?r^*e\right)$$

Untimed pattern matching: example

- Naive algorithm for pattern matching

c	r	e	p	e	s	$\in ?L\left(\{c\|i\|d\}^? r^* e\right)$
c						

Untimed pattern matching: example

- Naive algorithm for pattern matching

c	r	e	p	e	s	$\in ?L\left(\{c\|i\|d\}^? r^* e\right)$
c	r					

Untimed pattern matching: example

- Naive algorithm for pattern matching

c	r	e	p	e	s	$\in ?L\left(\{c\|i\|d\}^{?}r^{*}e\right)$
c	r	e				

Untimed pattern matching: example

- Naive algorithm for pattern matching

c	r	e	p	e	s	$\in ? L\left(\{c \mid i \mid d\}^? r^* e\right)$
c	r	e				$\surd$

Untimed pattern matching: example

- Naive algorithm for pattern matching

c	r	e	p	e	s	$\in ? L\left(\{c \mid i \mid d\}^? r^* e\right)$
c	r	e				$\surd$
	r					

Untimed pattern matching: example

- Naive algorithm for pattern matching

c	r	e	p	e	s	$\in ? L\left(\{c \mid i \mid d\}^? r^* e\right)$
c	r	e				$\surd$
	r	e				

Untimed pattern matching: example

- Naive algorithm for pattern matching

c	r	e	p	e	s	$\in ? L\left(\{c \vert i \vert d\}^? r^* e\right)$
c	r	e				$\surd$
	r	e				$\surd$

Untimed pattern matching: example

- Naive algorithm for pattern matching

c	r	e	p	e	s	$\in ? L\left(\{c \mid i \mid d\}^? r^* e\right)$
c	r	e				$\surd$
	r	e				$\surd$
		e				

Untimed pattern matching: example

- Naive algorithm for pattern matching

c	r	e	p	e	s	$\in ? L\left(\{c \mid i \mid d\}^? r^* e\right)$
c	r	e				$\surd$
	r	e				$\surd$
		e				$\surd$

Untimed pattern matching: example

- Naive algorithm for pattern matching

c	r	e	p	e	s	$\in ? L\left(\{c \mid i \mid d\}^? r^* e\right)$
c	r	e				$\surd$
	r	e				$\surd$
		e				$\surd$
			p			

Untimed pattern matching: example

- Naive algorithm for pattern matching

c	r	e	p	e	s	$\in ? L\left(\{c \mid i \mid d\}^? r^* e\right)$
c	r	e				$\surd$
	r	e				$\surd$
		e				$\surd$
			p			$\times$

Untimed pattern matching: example

- Naive algorithm for pattern matching

c	r	e	p	e	s	$\in ? L\left(\{c\|i\|d\}^? r^* e\right)$
c	r	e				$\surd$
	r	e				$\surd$
		e				$\surd$
			p			$\times$
				e		

Untimed pattern matching: example

- Naive algorithm for pattern matching

c	r	e	p	e	s	$\in ? L\left(\{c\|i\|d\}^? r^* e\right)$
c	r	e				$\surd$
	r	e				$\surd$
		e				$\surd$
			p			$\times$
				e		$\surd$

Untimed pattern matching: example

- Naive algorithm for pattern matching

c	r	e	p	e	s	$\in ? L\left(\{c\|i\|d\}^? r^* e\right)$
c	r	e				$\surd$
	r	e				$\surd$
		e				$\surd$
			p			$\times$
				e		$\surd$
					s	

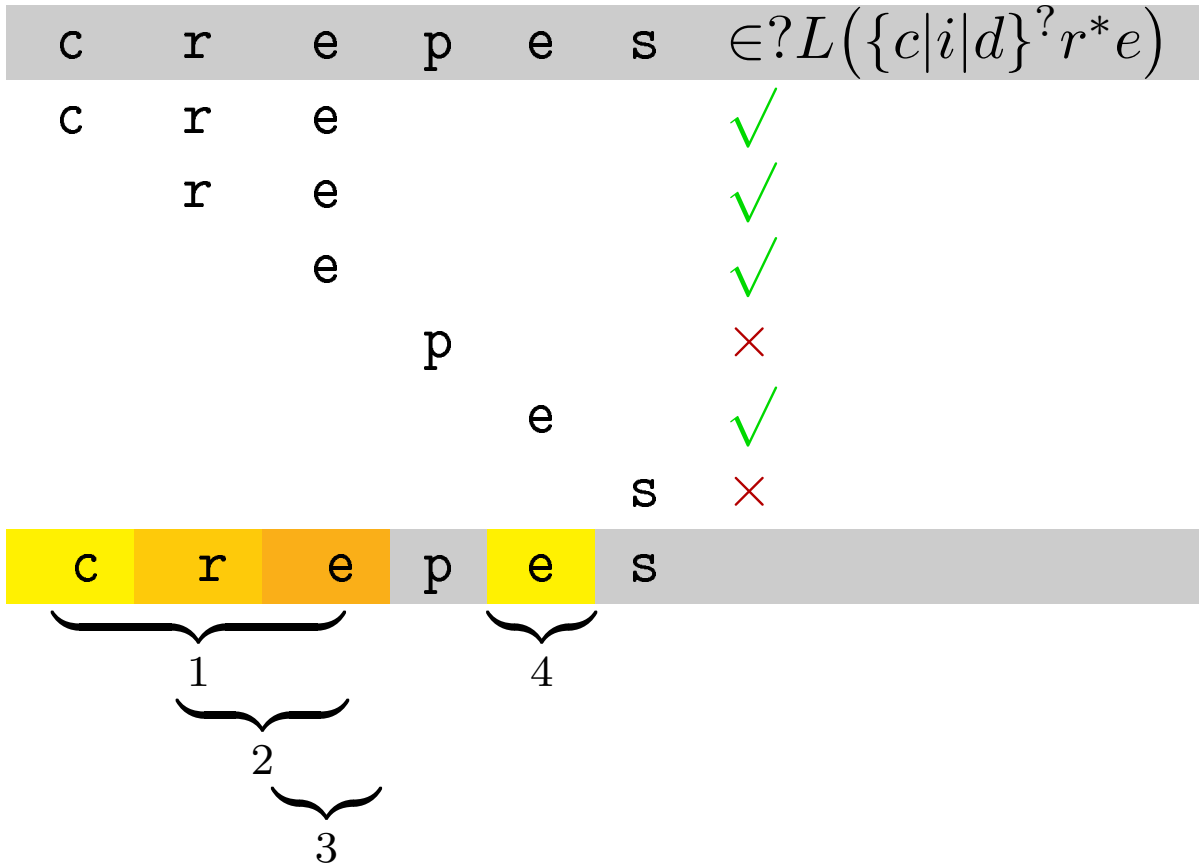
Untimed pattern matching: example

- Naive algorithm for pattern matching

c	r	e	p	e	s	$\in ? L\left(\{c\|i\|d\}^? r^* e\right)$
c	r	e				$\surd$
	r	e				$\surd$
		e				$\surd$
			p			$\times$
				e		$\surd$
					s	$\times$

Untimed pattern matching: example

- Naive algorithm for pattern matching



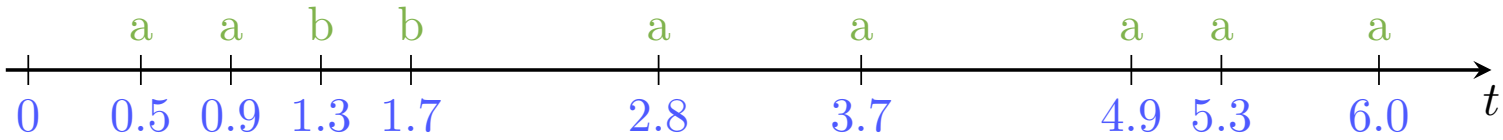
Timed pattern matching: timed word

Timed word

[Alur and Dill, 1994]

$=$

sequence of actions and timestamps



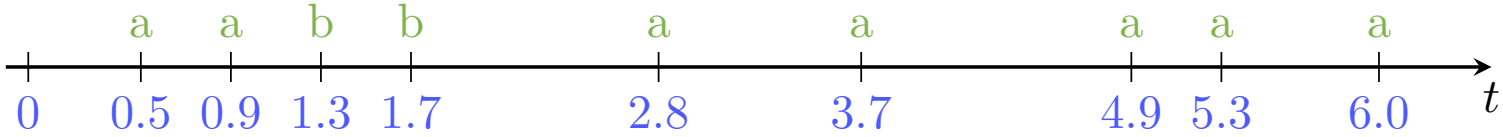
Timed pattern matching: timed word

Timed word

[Alur and Dill, 1994]

=

sequence of actions and timestamps

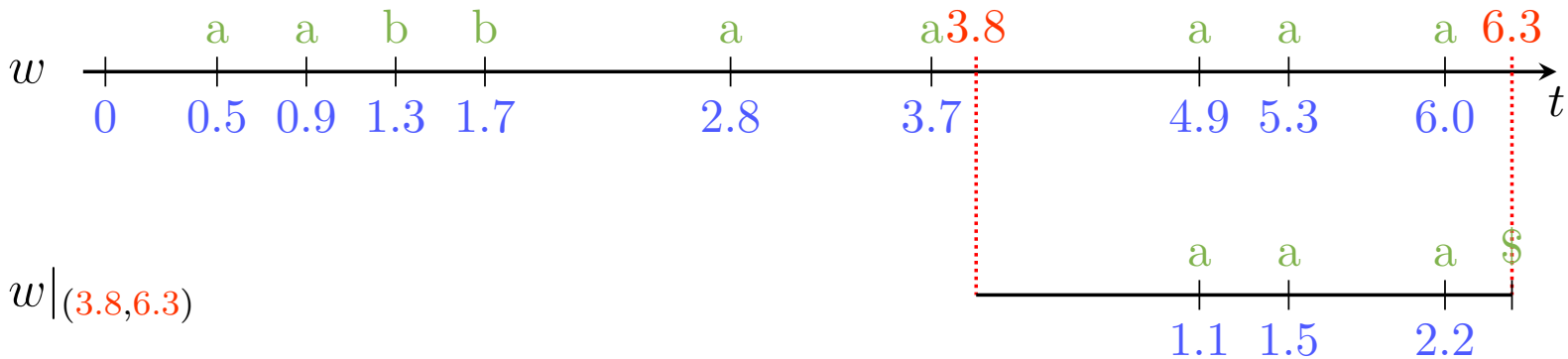


Timed word segment

[Waga et al., 2016]

=

projection of a segment of the timed word onto a given interval



Timed pattern matching: timed automaton

How to express a (timed) property on a log?

Example

"At least 1 time unit after the start of the segment, a is observed.
Then, within strictly less than 1 time unit, another a is observed.
Then, within strictly less than 1 time unit, another a is observed."

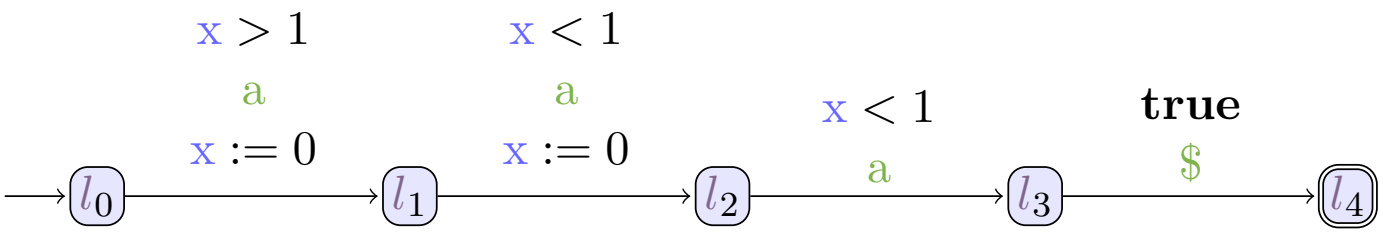
Timed pattern matching: timed automaton

How to express a (timed) property on a log?

Example

"At least 1 time unit after the start of the segment, a is observed.
Then, within strictly less than 1 time unit, another a is observed.
Then, within strictly less than 1 time unit, another a is observed."

A solution: timed automata [Alur and Dill, 1994]



- expressive
- well-studied
- supported by well-established model-checkers

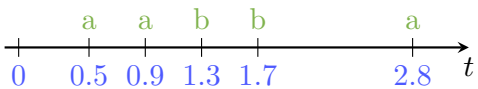
Timed pattern matching: principle

Timed pattern matching

- Inputs

A log

(timed word)

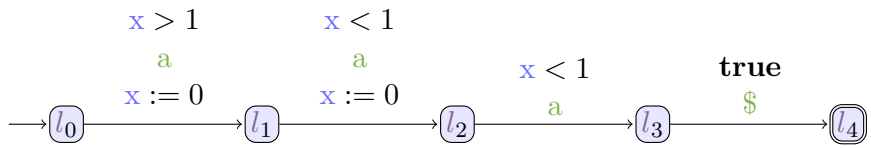


A property

usually a specification of faults

(timed automaton)

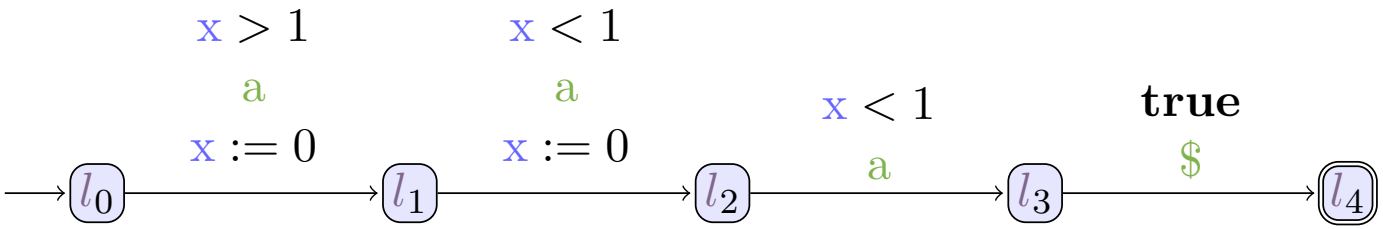
[Alur and Dill, 1994]



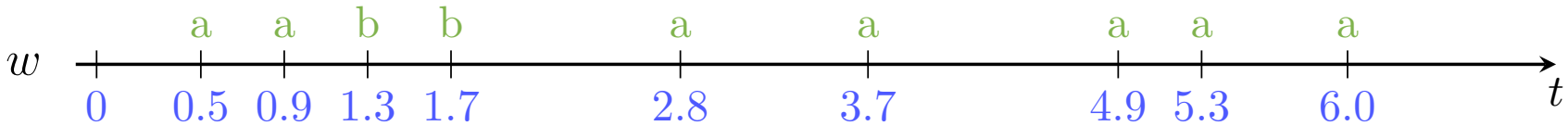
- Output
  - The set of time intervals where faults are detected
    $\Rightarrow$ Set of matching intervals $\{(t, t') \mid w|_{(t,t')} \in \mathcal{L}(\mathcal{A})\}$

Timed pattern matching: example

Our property:

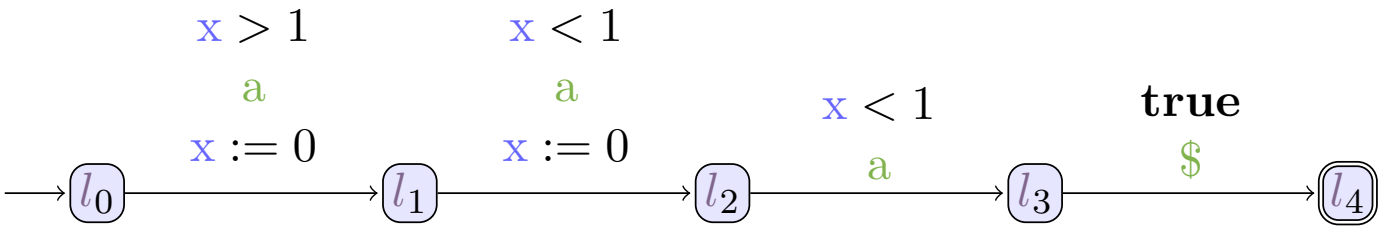


Our log:

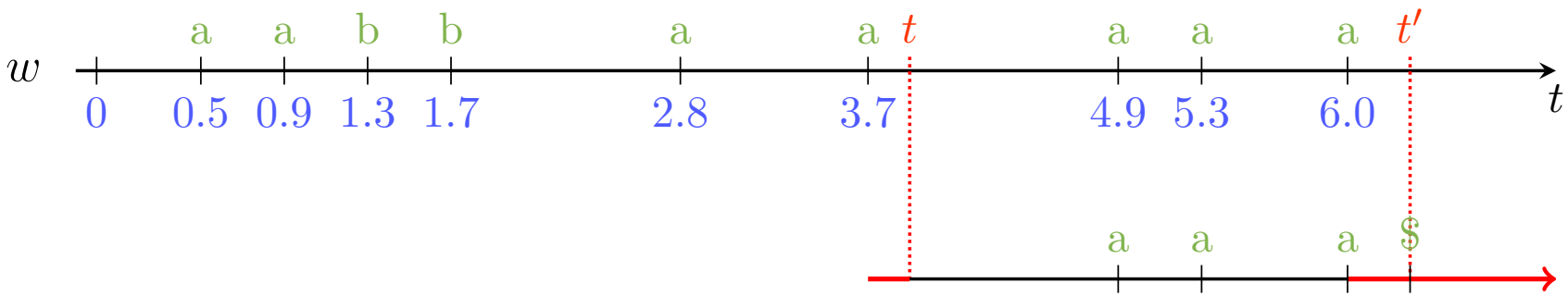


Timed pattern matching: example

Our property:

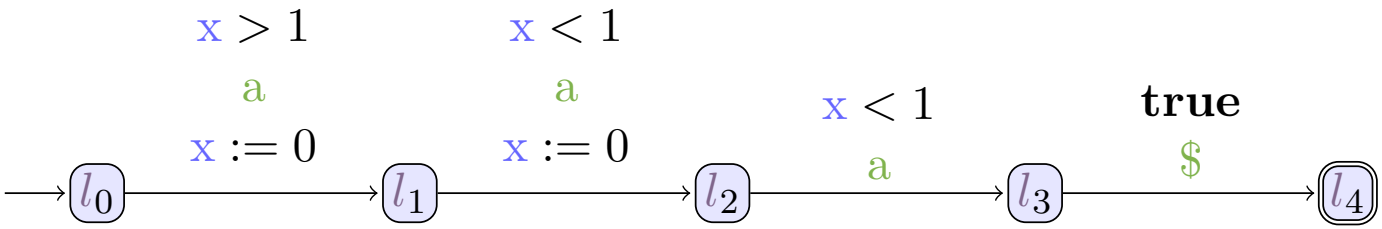


Our log:

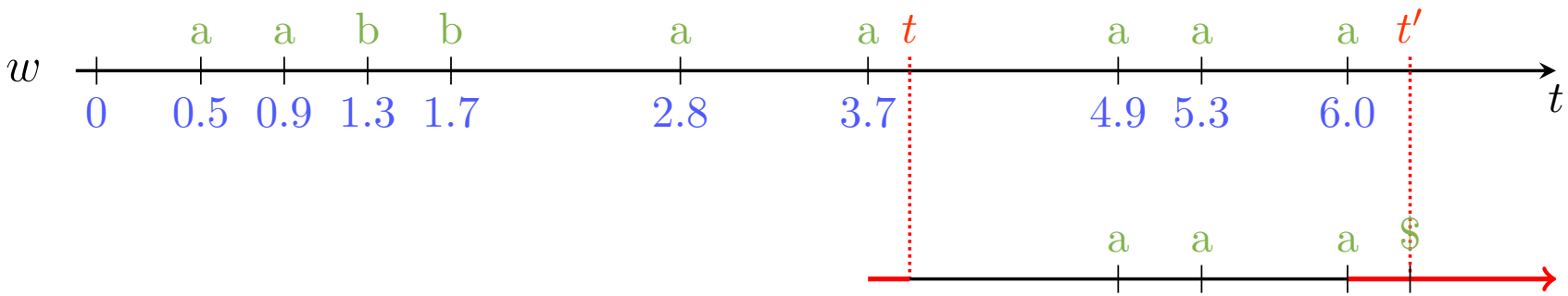


Timed pattern matching: example

Our property:



Our log:



Set of matching intervals:

$$\{(t,t') \mid w|_{(t,t')} \in \mathcal{L}(\mathcal{A})\} = \{(t,t') \mid t \in (3.7, 3.9), t' \in [6.0, \infty)\}$$

Goal: Extend timed pattern matching for uncertainty

Challenges

- The property may not be known with full certainty:
  - Detect a periodic event but without knowing the period
    - “is the user visiting this place more or less periodically?” (without knowing the actual period)
- Optimization problems
  - Find minimal/maximal timings for which some property holds
    - “what are the minimum/maximum intervals without visiting this shop”?

Goal: Extend timed pattern matching for uncertainty

Challenges

- The property may not be known with full certainty:
  - Detect a periodic event but without knowing the period
    - "is the user visiting this place more or less periodically?" (without knowing the actual period)
- Optimization problems
  - Find minimal/maximal timings for which some property holds
    - "what are the minimum/maximum intervals without visiting this shop"?

Objective

Find intervals of time and values of parameters for which a property holds

Outline

Methodology

Main idea

Use parametric timed model checking

- Formalism: parametric timed automata [Alur et al., 1993]
- Technique: parameter synthesis
- Software: IMITATOR [André et al., 2012]

Methodology

Main idea

Use parametric timed model checking

- Formalism: parametric timed automata [Alur et al., 1993]
- Technique: parameter synthesis
- Software: IMITATOR [André et al., 2012]

Methodology step by step

1. Encode the property using a PTA
2. Add two parameters t and t'
3. Apply a (mild) transformation to the property PTA
4. Transform the timed word into a PTA
5. Perform the composition of both PTA
6. Apply reachability synthesis to the product

Methodology

Main idea

Use parametric timed model checking

- Formalism: parametric timed automata [Alur et al., 1993]
- Technique: parameter synthesis
- Software: IMITATOR [André et al., 2012]

Methodology step by step

1. Encode the property using a PTA
2. Add two parameters t and t'
3. Apply a (mild) transformation to the property PTA
4. Transform the timed word into a PTA
5. Perform the composition of both PTA
6. Apply reachability synthesis to the product

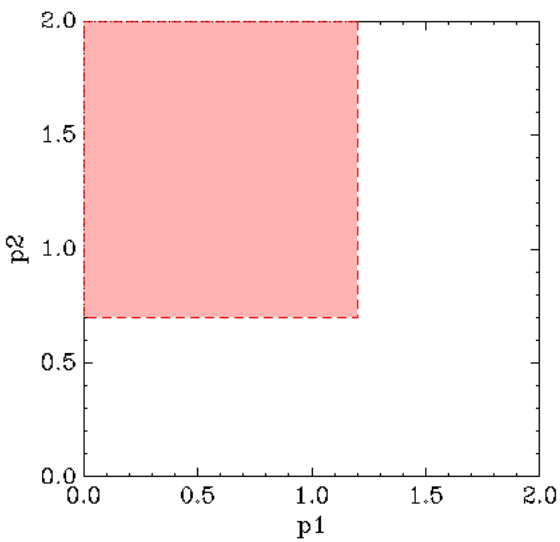
Teaser

Our method is scalable!

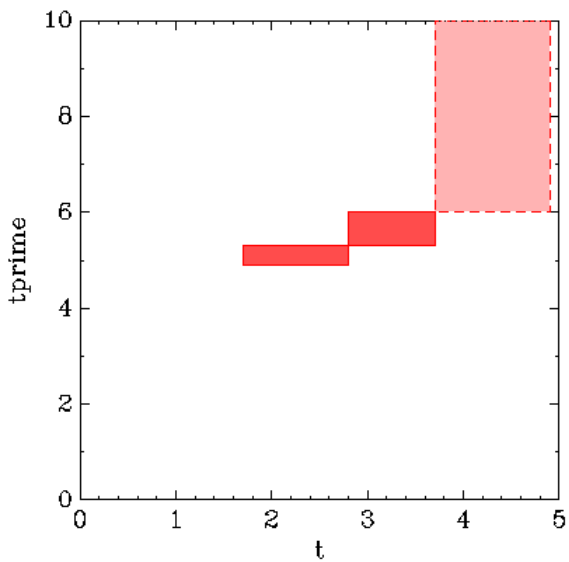
Example: graphical representation

$$
\begin{aligned}
& 1.7 < \mathrm{t} < 2.8 - \mathrm{p}_1 \wedge 4.9 \leq \mathrm{t}' < 5.3 \wedge \mathrm{p}_2 > 1.2 \\
\vee\ & 2.8 < \mathrm{t} < 3.7 - \mathrm{p}_1 \wedge 5.3 \leq \mathrm{t}' < 6 \wedge \mathrm{p}_2 > 1.2 \\
\vee\ & 3.7 < \mathrm{t} < 4.9 - \mathrm{p}_1 \wedge \mathrm{t}' \geq 6 \wedge \mathrm{p}_2 > 0.7
\end{aligned}
$$

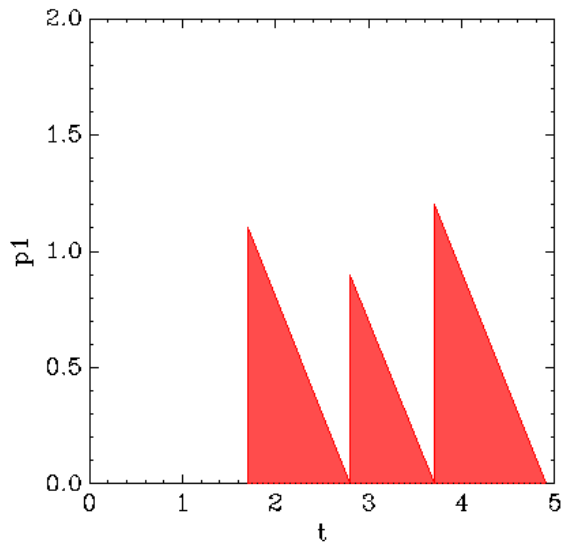
Projections in 2 dimensions:



On p_1 and p_2



On t and t'



On t and p_1

Outline

Case study 1: GEAR (description)

Monitoring the gear change of an automatic transmission system

- Obtained by simulation of the Simulink model of an automatic transmission system [Hoxha et al., 2014]
- S-TaLiRo [Annpureddy et al., 2011] used to generate an input to this model (generates a gear change signal that is fed to the model)
- Gear chosen from $\{g_1, g_2, g_3, g_4\}$
- Generated gear change recorded in a timed word

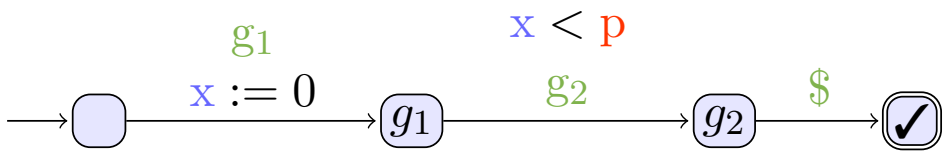
Property

"If the gear is changed to 1, it should not be changed to 2 within p seconds."

This condition is related to the requirement ϕ_5^{AT} proposed in [Hoxha et al., 2014] (the nominal value for p in [Hoxha et al., 2014] is 2).

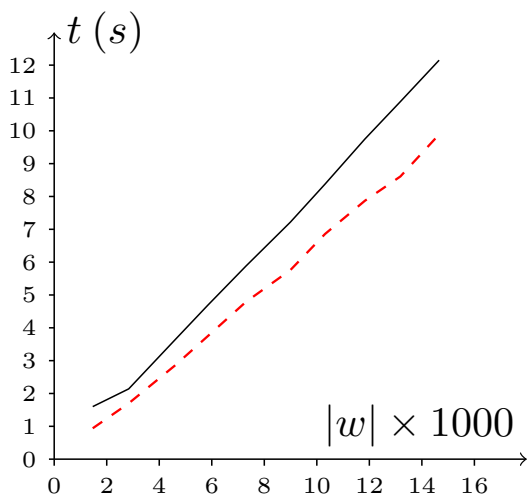
Case study 1: GEAR (experiments)

Property: "If the gear is changed to 1, it should not be changed to 2 within p seconds."



Experiments data:

Model		PTPM				PTPM_{opt}	
Length	Time frame	States	Matches	Parsing (s)	Comp. (s)	States	Comp. (s)
1,467	1,000	4,453	379	0.02	1.60	3,322	0.94
2,837	2,000	8,633	739	0.33	2.14	6,422	1.70
4,595	3,000	14,181	1,247	0.77	3.63	10,448	2.85
5,839	4,000	17,865	1,546	1.23	4.68	13,233	3.74
7,301	5,000	22,501	1,974	1.94	5.88	16,585	4.79
8,995	6,000	27,609	2,404	2.96	7.28	20,413	5.76
10,316	7,000	31,753	2,780	4.00	8.38	23,419	6.86
11,831	8,000	36,301	3,159	5.39	9.75	26,832	7.87
13,183	9,000	40,025	3,414	6.86	10.89	29,791	8.61
14,657	10,000	44,581	3,816	8.70	12.15	33,141	9.89



PTPM_{opt}: alternative procedure to find the minimum/maximum value of a parameter along the log

Case study 2: ACCEL (description)

Monitoring the acceleration of an automated transmission system

- Also obtained by simulation from the Simulink model of [Hoxha et al., 2014]
- (discretized) value of three state variables recorded in the log:
  - engine RPM (discretized to "high" and "low" with a certain threshold)
  - velocity (discretized to "high" and "low" with a certain threshold)
  - 4 gear positions

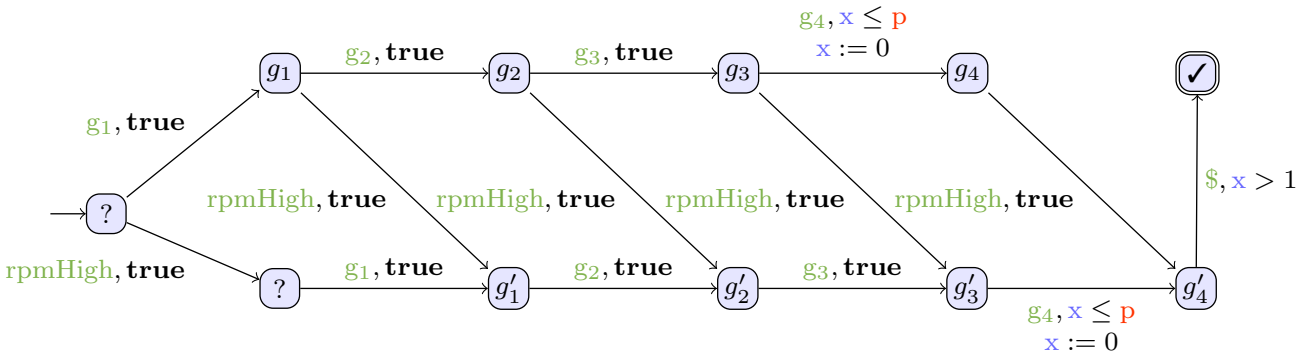
Property

"If a gear changes from 1 to 2, 3, and 4 in this order in p seconds and engine RPM becomes large during this gear change, then the velocity of the car must be sufficiently large in one second."

This condition models the requirement ϕ_8^{AT} proposed in [Hoxha et al., 2014] (the nominal value for p in [Hoxha et al., 2014] is 10).

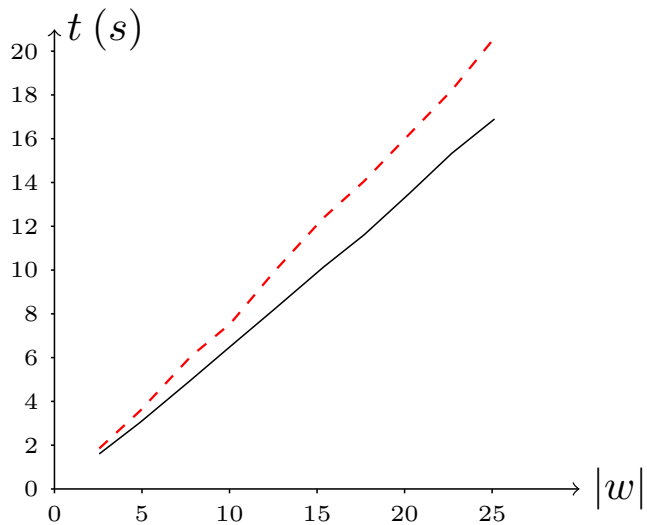
Case study 2: ACCEL (experiments)

Property: "If a gear changes from 1 to 2, 3, and 4 in this order in p seconds and engine RPM becomes large during this gear change, then the velocity of the car must be sufficiently large in one second."



Experiments data:

Model		PTPM				PTPM_{opt}	
Length	**Time frame**	**States**	**Matches**	**Parsing (s)**	**Comp. (s)**	**States**	**Comp. (s)**
2,559	1,000	6,504	2	0.27	1.60	6,502	1.85
4,894	2,000	12,429	2	0.86	3.04	12,426	3.57
7,799	3,000	19,922	7	2.21	4.98	19,908	6.06
10,045	4,000	25,520	3	3.74	6.51	25,514	7.55
12,531	5,000	31,951	9	6.01	8.19	31,926	9.91
15,375	6,000	39,152	7	9.68	10.14	39,129	12.39
17,688	7,000	45,065	9	13.40	11.61	45,039	14.06
20,299	8,000	51,660	10	18.45	13.52	51,629	16.23
22,691	9,000	57,534	11	24.33	15.33	57,506	18.21
25,137	10,000	63,773	13	31.35	16.90	63,739	20.61



Outline

Summary

- New original method to monitor logs of real-time systems
- Methodology: parametric timed model checking
- Applications: automotive industry
  - Linear in the size of the log
  - Able to handle logs of dozens of thousands of events
    $\Rightarrow$ scalable

Summary

- New original method to monitor logs of real-time systems
- Methodology: parametric timed model checking
- Applications: automotive industry
  - Linear in the size of the log
  - Able to handle logs of dozens of thousands of events
    $\Rightarrow$ scalable
- An ~~offline~~ online algorithm
  - We believe our algorithm is in fact essentially online
    - No need for the whole log to start the analysis
    - The word could be fed to IMITATOR in an incremental manner
  - But the speed may need to be improved further

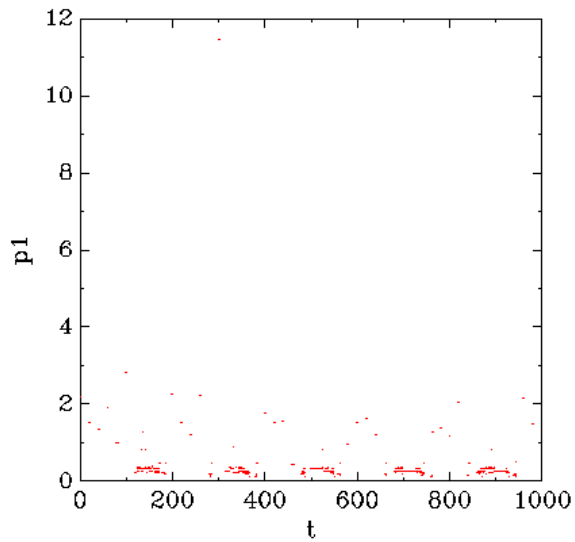
Perspectives

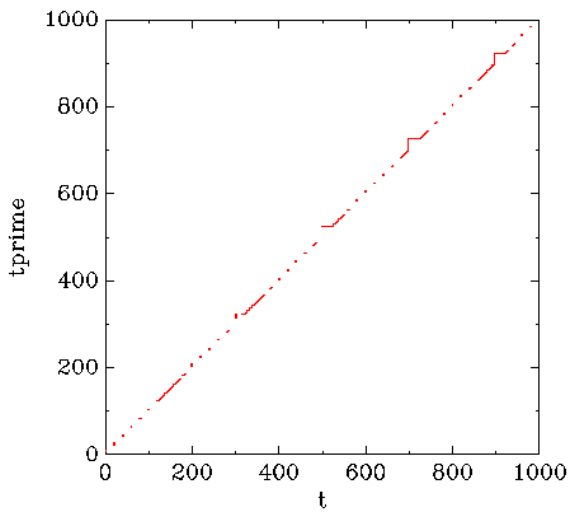
- Extensions
  - Extend to continuous-domain data and symbolic monitoring [Waga et al., 2019]
    - "Exhibit time frames t and customers c such that customer c withdrew more than half of the total money of all customers during time frame t?"
- Graphical representation and interpretation
  - How to interpret dozens of thousands of matches?

Perspectives

- Extensions
  - Extend to continuous-domain data and symbolic monitoring [Waga et al., 2019]
    - "Exhibit time frames t and customers c such that customer c withdrew more than half of the total money of all customers during time frame t?"

- Graphical representation and interpretation
  - How to interpret dozens of thousands of matches?





Bibliography

References I

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., 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 *LNCS*, pages 33–36. Springer.

Annpureddy, Y., Liu, C., Fainekos, G. E., and Sankaranarayanan, S. (2011).
S-TaLiRo: A tool for temporal logic falsification for hybrid systems.
In Abdulla, P. A. and Leino, K. R. M., editors, *TACAS*, volume 6605 of *LNCS*, pages 254–257. Springer.

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.

Waga, M., Akazaki, T., and Hasuo, I. (2016).
A Boyer-Moore type algorithm for timed pattern matching.
In Fränzle, M. and Markey, N., editors, *FORMATS*, volume 9884 of *LNCS*, pages 121–139. Springer.

References II

Waga, M., André, É., and Hasuo, I. (2019).
Symbolic monitoring against specifications parametric in time and data.
In Dillig, I. and Tasiran, S., editors, *CAV, Part I*, volume 11561 of *LNCS*, pages 520–539. Springer.

Licensing

Source of the graphics used I

Title: 1960 Citroen DS19
Author: Joc281
Source: https://en.wikipedia.org/wiki/File:800px_1973_377_Citroen_DS19_automatically_guided_motor
License: CC by-sa 3.0

Title: A Cartoon Businessman Reading A Text Message
Author: Vector Toons
Source: https://en.wikipedia.org/wiki/File:800px_1973_377_Citroen_DS19_automatically_guided_motor
License: CC by-sa 4.0

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

Author: **Étienne André**