A benchmark library for parametric timed model checking

Étienne André

A benchmark library for parametric timed model checking*

Étienne André[1,2,3][0000−0001−8473−9555]

[1] Université Paris 13, LIPN, CNRS, UMR 7030, F-93430, Villetaneuse, France
[2] JFLI, CNRS, Tokyo, Japan
[3] National Institute of Informatics, Japan

Abstract. Verification of real-time systems involving hard timing constraints and concurrency is of utmost importance. Parametric timed model checking allows for formal verification in the presence of unknown timing constants or uncertainty (e. g., imprecision for periods). With the recent development of several techniques and tools to improve the efficiency of parametric timed model checking, there is a growing need for proper benchmarks to test and compare fairly these tools. We present here a benchmark library for parametric timed model checking made of benchmarks accumulated over the years. Our benchmarks include academic benchmarks, industrial case studies and examples unsolvable using existing techniques.

Keywords: case studies · model checking · parameter synthesis · parametric timed automata

1 Introduction

Verification of real-time systems involving hard timing constraints and concurrency is of utmost importance, and is now recognized in standards such as the DO-178C, that allows formal methods without addressing specific process requirements. Model checking is a popular model-based technique that formally verifies whether a model satisfies a property. Parametric timed model checking significantly enhances model checking by allowing its application earlier in the design phase, when timing constants may not be known yet. In addition, it is possible to verify systems in the presence of uncertainty, e. g., when some periods are known with some limited precision. This is the case of Thales' FMTV[1] challenge 2014 where the system was characterized with uncertain but *constant* periods, that rules out the use of non-parametric timed model checking.

* This is the author version of the manuscript of the same name published in the proceedings of the Sixth International Workshop on Formal Techniques for Safety-Critical Systems (FTSCS 2018). This work is partially supported by the ANR national research program PACS (ANR-14-CE28-0002) and by ERATO HASUO Metamathematics for Systems Design Project (No. JPMJER1603), JST.

[1] "Formal Methods for Timing Verification Challenge", in the WATERS workshop: http://waters2015.inria.fr/

Popular formalism for parametric timed model checking include parametric timed automata (PTAs) [AHV93] and parametric time Petri nets [TLR09].

Several tools support parameters, such as HyTech [HHWT95] (parametric hybrid automata), Romeo [LRST09] (parametric time Petri nets), IMITATOR [AFKS12] (parametric timed automata), PSyHCoS [ALS+13] (parametric stateful timed CSP), or `Symrob` (robustness for timed automata) [San15]. In addition, several tools support the larger class of *hybrid automata*, such as `PHAVer` [Fre08] or `SpaceEx` [FLGD+11] and, while not explicitly supporting parameters, can encode them.[2] Recently, a growing number of analyses and techniques were proposed to analyze parametric timed models (mainly PTAs) such as SMT-based techniques [KP12], integer hull abstractions [JLR15], corner-point abstractions [BBLS15], distributed verification [ACN15], NDFS-based synthesis [NPvdP18], machine learning [AL17,LSGA17], etc. However, despite some case studies informally shared between these works, there is a lack of a common basis to compare new tools and techniques in a fair manner. Without a stable list of benchmarks publicly available, it is difficult to assess the efficiency of a new algorithm.

Contribution. We present here a library of benchmarks containing academic and industrial case studies collected in the past few years from academic papers and industrial collaborations. In addition, a focus is made on (possibly toy) examples known to be unsolvable using current state-of-the-art techniques, with the hope to encourage the development of new techniques to solve them. Benchmarks are available online in the IMITATOR input format, and distributed using the GNU General Public License.

Related libraries. The library most related to ours is that by Chen *et al.*, that proposes a suite of benchmarks for hybrid systems [CSBM+15]. However, it aims at analyzing hybrid systems, which are strictly more expressive than PTAs in theory, and incomparable in practice, as most hybrid systems do not feature timing parameters. In addition, that benchmark suite focuses only on reachability properties. Finally and most importantly, it does not focus on parameters, and the benchmarks are non-parametric. In contrast, our library focuses on parametric timed benchmarks, with various types of properties.

Another interesting library is that by Hoxha, Abbas, and Fainekos [HAF14], that offers Matlab/Simulink models of automotive systems. However, it does not aim specifically at parametric timed model checking; two of our benchmarks originally partially come from the aforementioned library [HAF14].

2 IMITATOR parametric timed automata

Parametric timed automata extend finite-state automata with clocks, i. e., real-valued variables evolving at the same rate. Clocks can be reset along transitions,

[2] In a hybrid automaton, a parameter is a variable that can evolve for an arbitrary amount of time at rate 1, and is then "frozen" (rate 0).

(a) Reaching l_1 if $p \in \{\frac{1}{n} \mid n \in \mathbb{N}_{>0}\}$ **(b)** Reaching l_1 if $p \in \mathbb{N}$

Fig. 1: Examples of PTAs

and can be compared to constants or parameters (integer- or rational-valued) along transitions ("guards") or in locations ("invariants"). IMITATOR parametric timed automata extend PTAs [AHV93] with some useful features such as synchronization between components, stopwatches (i. e., the ability to stop the elapsing of some clocks [CL00]), presence of parametric linear terms in guards, invariants and resets, shared global rational-valued variables, etc.

Example 1. Consider the PTA in Fig. 1a, containing two locations l_0 and l_1, two clocks x and y, and one parameter p. The self-loop on l_0 can be taken whenever $x = p$ holds, and resets x, i. e., can be taken every p time units. In addition, initially, as $x = y = 0$ and clocks evolve at the same rate, the transition guarded by $y = 1 \wedge x = 0$ cannot be taken. Observe that, if $p = 1$, then the transition to l_1 can be taken after exactly one loop on l_0. If $p = \frac{1}{2}$, then the transition to l_1 can be taken after exactly two loops. In fact, the set of valuations for which l_1 is reachable is exactly $\{i \mid i = \frac{1}{n}, n > 0 \wedge n \in \mathbb{N}\}$.

L/U-PTAs. Lower-bound/upper-bound parametric timed automata (L/U-PTAs) [HRSV02] restrict the use of parameters: parameters must be partitioned between lower-bound parameters (always compared with clocks as lower bounds, i. e., $p \leq x$ or $p < x$) and upper-bound parameters. L/U-PTAs enjoy monotonicity properties and, while the full class of PTAs is highly undecidable [And18], L/U-PTAs enjoy some decidability results [HRSV02,BL09,ALR18b]. U-PTAs [BL09,ALR18a] are L/U-PTAs with only upper-bound parameters.

3 The benchmark library

3.1 Categories

Our benchmarks are classified into three main categories:

1. academic benchmarks, studied in a range of papers: a typical example is the Fischer mutual exclusion protocol;
2. industrial case studies, which correspond to a concrete problem solved (or not) in an industrial environment;

3. examples famous for being unsolvable using state-of-the-art techniques; for some of them, a solution may be computed by hand, but existing automated techniques are not capable of computing it. This is the case of the PTA in Fig. 1a, as a human can very easily solve it, while (to the best of our knowledge) no tool is able to compute this result automatically.

Remark 1. Our library contains a fourth category: education benchmarks, that consist of generally simple case studies that can be used for *teaching*. This category contains toy examples such as coffee machines. We omit this category from this paper as these benchmarks generally have a limited interest performance wise.

The domain of the benchmarks are hardware asynchronous circuits, communication or mutual exclusion protocols, real-time systems ("RTS") and schedulability problems, parametric timed pattern matching ("PTPM"), train-gate-controllers models ("TGC"), etc.

In addition, we use the following classification criteria:

- number of variables: clocks, parameters, locations, automata;
- whether the benchmark (in the provided version) is easily *scalable*, i. e., whether one can generate a large number of instances; for example, protocols often depend on the number of participants, and can therefore be scaled accordingly;
- presence of shared rational-valued variables;
- presence of stopwatches;
- presence of location invariants, as some works (e. g., [AHV93,ALR18a]) exclude them;
- whether the benchmark meets the L/U assumption.

3.2 Properties

We consider the three following main properties:

reachability / safety: synthesize parameter valuations for which a given state of the system (generally a location, but possibly a constraint on variables) must be reachable / avoided (see e. g., [JLR15]).

optimal reachability: same as reachability, but with an optimization criterion: some parameters (or the time) should be minimized or maximized.

unavoidability: synthesize parameter valuations for which all runs must always eventually reach a given state (see e. g., [JLR15]).

robustness: synthesize parameter valuations preserving the discrete behavior (untimed language) w.r.t. to a given valuation (see e. g., [ACEF09,San15]).

In addition, we include some recent case studies of parametric timed pattern matching ("PTPM" hereafter), i. e., being able to decide for which part of a log and for which values of parameters does a parametric property holds on that log [AHW18]. Finally, a few more case studies have *ad-hoc* properties (liveness, properties expressed using observers [ABBL98,And13], etc.), denoted "Misc." later on.

3.3 Presentation

The benchmark library comes in the form of a Web page that classifies models and is available at `https://www.imitator.fr/library.html`.

The library is made of a list of a set of *benchmarks*. Each benchmark may have different *models*: for example, `Flip-flop` comes with three models, one with 2 parameters, one with 5, and one with 12 parameters. Similarly, some `Fischer` benchmarks come with several models, each of them corresponding to a different number of processes. Finally, each model comes with one or more *properties*. For example, for `Fischer`, one can either run safety synthesis, or evaluate the robustness of a given reference parameter valuation.

The first version of the library contains 34 benchmarks with 80 different models and 122 properties.

3.4 Performance

We present a selection of the library in Table 1. Not all benchmarks are given; in addition, most benchmarks come with several models and several properties, omitted here for space concern. We give from left to right the number of automata, of clocks, of parameters, of discrete variables, whether the model is an L/U-PTA, a U-PTA or a regular PTA, whether it features invariants and stopwatches, the kind of property, and a computation time on an Intel i7-7500U CPU @ 2.70GHz with 8 GiB running Linux Mint 18.

"T.O." denotes time-out (after 300 s). "?" denotes unsolvable, because no such algorithm is implemented in existing tools. "HS" denotes time-out but human-solvable: e. g., for Fischer, one knows the correctness constraint independently of the number of processes, but tools may fail to compute it. This is also the case of the toy PTAs in Figs. 1a and 1b.

Despite time-out, some case studies come with a partial result: either because IMITATOR is running reachability-synthesis ("EFsynth" [JLR15]) which can output a partial result when interrupted before completion, or because some other methods can output some valuations. For example, for `ProdCons`, IMITATOR is unable to synthesize a constraint; however, in the original work [KP12], some punctual valuations (non-symbolic) are given.

Robustness case studies are not part of Table 1, but are included in the online library.

4 Perspectives

Syntax. So far, all benchmarks use the IMITATOR input format; in addition, only if the benchmark comes from another model checker (e. g., a HyTech or Uppaal model), it also comes with its native syntax. In a near future, we plan to propose a translation to Uppaal timed automata; however, some information will be lost as Uppaal does not allow parameters, and supports stopwatches in a limited manner. A future work will be to propose other syntaxes, or a normalized syntax for parametric timed model checking benchmarks.

Table 1: A selection from the benchmark library

Benchmark	Ref	Domain	Scal.	$\lvert\mathcal{A}\rvert$	$\lvert\mathbb{X}\rvert$	$\lvert\mathbb{P}\rvert$	$\lvert\mathbb{V}\rvert$	L/U	Inv	SW	Prop.	Time
Academic												
`And-Or`	[CC05]	Circuit	×	4	4	12	0	–	√	×	Misc.	3.01
`CSMA/CD`	[KNSW07]	Protocol	√	3	3	3	0	–	√	×	Unavoid.	?
`Fischer-AHV93`	[AHV93]	Protocol	√	3	2	4	0	L/U	×	×	Safety	0.04
`Fischer-HRSV02:3`	[HRSV02]	Protocol	√	3	3	4	1	L/U	√	×	Safety	HS
`Flip-flop:2`	[CC07]	Circuit	×	5	5	2	0	U	√	×	Misc.	0.04
`Flip-flop:12`	[CC07]	Circuit	×	5	5	12	0	U	√	×	Misc.	23.07
`idle-time-sched:3`	[LSAF14]	RTS	√	8	13	2	3	U	√	√	Safety	1.49
`idle-time-sched:5`	[LSAF14]	RTS	√	12	21	2	0	U	√	√	Safety	14.61
`Jobshop:3-4`	[AM01]	Sched.	√	2	3	12	4	–	√	×	Opt. reach.	5.58
`Jobshop:4-4`	[AM01]	Sched.	√	4	4	16	4	–	√	×	Opt. reach.	T.O.
`NP-FPS-3tasks:50-0`	[JLR13]	RTS	×	4	6	2	0	–	√	×	Safety	1.03
`NP-FPS-3tasks:100-2`	[JLR13]	RTS	×	4	6	2	0	–	√	×	Safety	65.23
`SSLAF14-1`	[PGGH98,SSL+13]	RTS	×	7	16	2	2	–	√	√	Safety	0.33
`SSLAF14-2`	[WTVL06,SSL+13]	RTS	×	6	14	2	4	–	√	√	Safety	T.O.
`ProdCons:2-3`	[KP12]	Prod.-cons.	√	5	5	6	0	L/U	√	×	Reach.	T.O.
`train-AHV93`	[AHV93]	TGC	×	3	3	6	0	L/U	×	×	Safety	0.01
`WFAS`	[BBLS15]	Protocol	×	3	4	2	0	–	√	×	Safety	T.O.
Industrial												
`accel:1`	[HAF14,AHW18]	PTPM	√	2	2	3	0	–	√	×	PTPM	1.25
`accel:10`	[HAF14,AHW18]	PTPM	√	2	2	3	0	–	√	×	PTPM	12.67
`BRP`	[DKRT97]	Protocol	×	6	7	2	12	–	√	×	Safety	248.35
`FMTV:1A1`	[SAL15]	RTS	×	3	3	3	5	–	√	×	Opt. reach.	6.97
`FMTV:1A3`	[SAL15]	RTS	×	3	3	3	7	–	√	×	Opt. reach.	87.39
`FMTV:2`	[SAL15]	RTS	×	6	9	2	0	–	√	√	Opt. reach.	1.61
`gear:1`	[HAF14,AHW18]	PTPM	√	2	2	3	0	–	√	×	PTPM	0.77
`gear:10`	[HAF14,AHW18]	PTPM	√	2	2	3	0	–	√	×	PTPM	7.42
`RCP`	[CAS01]	Protocol	×	5	6	5	6	L/U	√	×	Reach.	1.07
`SIMOP:3`	[ACD+09]	Automation	×	5	8	3	0	–	√	×	Reach.	T.O.
`SPSMALL:2`	[CEFX09]	Circuit	×	11	11	2	0	–	√	×	Reach.	0.96
`SPSMALL:26`	[CEFX09]	Circuit	×	11	11	26	0	–	√	×	Reach.	T.O.
Toy												
`toy:n`	Fig. 1b	Toy	×	1	2	1	0	–	√	×	Reach.	HS
`toy:1/n`	Fig. 1a	Toy	×	1	2	1	0	U	√	×	Reach.	HS

Contributions and versioning. The library is aimed at being enriched with future benchmarks. Furthermore, it is collaborative, and is open to any willing contributor. A versioning system will be set up with the addition (or modification) of benchmarks in the future.

References

ABBL98. Luca Aceto, Patricia Bouyer, Augusto Burgueño, and Kim Guldstrand Larsen. The power of reachability testing for timed automata. In Vikraman Arvind and Ramaswamy Ramanujam, editors, *Proceedings of the 18th Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 1998)*, volume 1530 of *Lecture Notes in Computer Science*, pages 245–256. Springer, 1998.

ACD+09. Étienne André, Thomas Chatain, Olivier De Smet, Laurent Fribourg, and Silvain Ruel. Synthèse de contraintes temporisées pour une architecture d'automatisation en réseau. In Didier Lime and Olivier H. Roux, editors, *Actes du 7ème colloque sur la modélisation des systèmes réactifs (MSR 2009)*, volume 43 of *Journal Européen des Systèmes Automatisés*, pages 1049–1064. Hermès, November 2009.

ACEF09. Étienne André, Thomas Chatain, Emmanuelle Encrenaz, and Laurent Fribourg. An inverse method for parametric timed automata. *International Journal of Foundations of Computer Science*, 20(5):819–836, October 2009.

ACN15. Étienne André, Camille Coti, and Hoang Gia Nguyen. Enhanced distributed behavioral cartography of parametric timed automata. In Michael Butler, Sylvain Conchon, and Fatiha Zaïdi, editors, *Proceedings of the 17th International Conference on Formal Engineering Methods (ICFEM 2015)*, volume 9407 of *Lecture Notes in Computer Science*, pages 319–335. Springer, November 2015.

AFKS12. Étienne André, Laurent Fribourg, Ulrich Kühne, and Romain Soulat. IMITATOR 2.5: A tool for analyzing robustness in scheduling problems. In Dimitra Giannakopoulou and Dominique Méry, editors, *Proceedings of the 18th International Symposium on Formal Methods (FM 2012)*, volume 7436 of *Lecture Notes in Computer Science*, pages 33–36. Springer, August 2012.

AHV93. Rajeev Alur, Thomas A. Henzinger, and Moshe Y. Vardi. Parametric real-time reasoning. In S. Rao Kosaraju, David S. Johnson, and Alok Aggarwal, editors, *Proceedings of the twenty-fifth annual ACM symposium on Theory of computing (STOC 1993)*, pages 592–601, New York, NY, USA, 1993. ACM.

AHW18. Étienne André, Ichiro Hasuo, and Masaki Waga. Offline timed pattern matching under uncertainty. In Anthony Widjaja Lin and Jun Sun, editors, *Proceedings of the 23rd International Conference on Engineering of Complex Computer Systems (ICECCS 2018)*. IEEE, 2018. To appear.

AL17. Étienne André and Shang-Wei Lin. Learning-based compositional parameter synthesis for event-recording automata. In Ahmed Bouajjani and Silva Alexandra, editors, *Proceedings of the 37th IFIP WG 6.1 International Conference on Formal Techniques for Distributed Objects, Components, and Systems (FORTE 2017)*, volume 10321 of *Lecture Notes in Computer Science*, pages 17–32. Springer, 2017.

ALR18a. Étienne André, Didier Lime, and Mathias Ramparison. TCTL model checking lower/upper-bound parametric timed automata without invariants. In David N. Jansen and Pavithra Prabhakar, editors, *Proceedings of the 16th International Conference on Formal Modeling and Analysis of Timed Systems (FORMATS 2018)*, volume 11022 of *Lecture Notes in Computer Science*, pages 1–17. Springer, 2018.

ALR18b. Étienne André, Didier Lime, and Mathias Ramparison. Timed automata with parametric updates. In Gabriel Juhás, Thomas Chatain, and Radu Grosu, editors, *Proceedings of the 18th International Conference on Application of Concurrency to System Design (ACSD 2018)*, pages 21–29. IEEE, 2018. To appear.

ALS+13. Étienne André, Yang Liu, Jun Sun, Jin Song Dong, and Shang-Wei Lin. PSyHCoS: Parameter synthesis for hierarchical concurrent real-time sys-

tems. In Natasha Sharygina and Helmut Veith, editors, *Proceedings of the 25th International Conference on Computer Aided Verification (CAV 2013)*, volume 8044 of *Lecture Notes in Computer Science*, pages 984–989, Heidelberg, Germany, July 2013. Springer.

AM01. Yasmina Abdeddaïm and Oded Maler. Job-shop scheduling using timed automata. In Gérard Berry, Hubert Comon, and Alain Finkel, editors, *Proceedings of the 13th International Conference on Computer Aided Verification (CAV 2001)*, volume 2102 of *Lecture Notes in Computer Science*, pages 478–492. Springer, 2001.

And13. Étienne André. Observer patterns for real-time systems. In Yang Liu and Andrew Martin, editors, *18th IEEE International Conference on Engineering of Complex Computer Systems (ICECCS 2013)*, pages 125–134. IEEE Computer Society, July 2013.

And18. Étienne André. What's decidable about parametric timed automata? *International Journal on Software Tools for Technology Transfer*, 2018. To appear.

BBLS15. Nikola Beneš, Peter Bezděk, Kim Gulstrand Larsen, and Jiří Srba. Language emptiness of continuous-time parametric timed automata. In Magnús M. Halldórsson, Kazuo Iwama, Naoki Kobayashi, and Bettina Speckmann, editors, *Proceedings of the 42nd International Colloquium on Automata, Languages, and Programming (ICALP 2015), Part II*, volume 9135 of *Lecture Notes in Computer Science*, pages 69–81. Springer, July 2015.

BL09. Laura Bozzelli and Salvatore La Torre. Decision problems for lower/upper bound parametric timed automata. *Formal Methods in System Design*, 35(2):121–151, 2009.

CAS01. Aurore Collomb-Annichini and Mihaela Sighireanu. Parameterized reachability analysis of the IEEE 1394 root contention protocol using TReX. In *Proceedings of the Real-Time Tools Workshop (RT-TOOLS 2001)*, 2001.

CC05. Robert Clarisó and Jordi Cortadella. Verification of concurrent systems with parametric delays using octahedra. In *Proceedings of the Fifth International Conference on Application of Concurrency to System Design (ACSD 2005)*, pages 122–131. IEEE Computer Society, 2005.

CC07. Robert Clarisó and Jordi Cortadella. The octahedron abstract domain. *Science of Computer Programming*, 64(1):115–139, 2007.

CEFX09. Rémy Chevallier, Emmanuelle Encrenaz-Tiphène, Laurent Fribourg, and Weiwen Xu. Timed verification of the generic architecture of a memory circuit using parametric timed automata. *Formal Methods in System Design*, 34(1):59–81, February 2009.

CL00. Franck Cassez and Kim Guldstrand Larsen. The impressive power of stopwatches. In Catuscia Palamidessi, editor, *Proceedings of the 11th International Conference on Concurrency Theory (CONCUR 2000)*, volume 1877 of *Lecture Notes in Computer Science*, pages 138–152. Springer, 2000.

CSBM+15. Xin Chen, Stefan Schupp, Ibtissem Ben Makhlouf, Erika Ábrahám, Goran Frehse, and Stefan Kowalewski. A benchmark suite for hybrid systems reachability analysis. In Klaus Havelund, Gerard J. Holzmann, and Rajeev Joshi, editors, *Proceedings of the 7th International Symposium NASA Formal Methods (NFM 2015)*, volume 9058 of *Lecture Notes in Computer Science*, pages 408–414. Springer, 2015.

DKRT97. Pedro R. D'Argenio, Joost-Pieter Katoen, Theo C. Ruys, and Jan Tretmans. The bounded retransmission protocol must be on time! In Ed Brinksma, editor, *Proceedings of the Third International Workshop on Tools and Algorithms for Construction and Analysis of Systems (TACAS 1997)*, volume 1217 of *Lecture Notes in Computer Science*, pages 416–431. Springer, 1997.

FLGD+11. Goran Frehse, Colas Le Guernic, Alexandre Donzé, Scott Cotton, Rajarshi Ray, Olivier Lebeltel, Rodolfo Ripado, Antoine Girard, Thao Dang, and Oded Maler. SpaceEx: Scalable verification of hybrid systems. In Ganesh Gopalakrishnan and Shaz Qadeer, editors, *Proceedings of the 23rd International Conference on Computer Aided Verification (CAV 2011)*, volume 6806 of *Lecture Notes in Computer Science*, pages 379–395. Springer, 2011.

Fre08. Goran Frehse. PHAVer: Algorithmic verification of hybrid systems past HyTech. *International Journal on Software Tools for Technology Transfer*, 10(3):263–279, May 2008.

HAF14. Bardh Hoxha, Houssam Abbas, and Georgios E. Fainekos. Benchmarks for temporal logic requirements for automotive systems. In Goran Frehse and Matthias Althoff, editors, *Proceedings of the 1st and 2nd International Workshops on Applied veRification for Continuous and Hybrid Systems (ARCH@CPSWeek 2014 / ARCH@CPSWeek 2015)*, volume 34 of *EPiC Series in Computing*, pages 25–30. EasyChair, 2014.

HHWT95. Thomas A. Henzinger, Pei-Hsin Ho, and Howard Wong-Toi. A user guide to HyTech. In Ed Brinksma, Rance Cleaveland, Kim Guldstrand Larsen, Tiziana Margaria, and Bernhard Steffen, editors, *Proceedings of the First International Workshop on Tools and Algorithms for Construction and Analysis of Systems (TACAS 1995)*, volume 1019 of *Lecture Notes in Computer Science*, pages 41–71. Springer, 1995.

HRSV02. Thomas Hune, Judi Romijn, Mariëlle Stoelinga, and Frits W. Vaandrager. Linear parametric model checking of timed automata. *Journal of Logic and Algebraic Programming*, 52-53:183–220, 2002.

JLR13. Aleksandra Jovanović, Didier Lime, and Olivier H. Roux. Integer parameter synthesis for timed automata. In Nir Piterman and Scott A. Smolka, editors, *Proceedings of the 19th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS 2013)*, volume 7795 of *Lecture Notes in Computer Science*, pages 401–415. Springer, 2013.

JLR15. Aleksandra Jovanović, Didier Lime, and Olivier H. Roux. Integer parameter synthesis for timed automata. *IEEE Transactions on Software Engineering*, 41(5):445–461, 2015.

KNSW07. Marta Z. Kwiatkowska, Gethin Norman, Jeremy Sproston, and Fuzhi Wang. Symbolic model checking for probabilistic timed automata. *Information and Computation*, 205(7):1027–1077, 2007.

KP12. Michał Knapik and Wojciech Penczek. Bounded model checking for parametric timed automata. *Transactions on Petri Nets and Other Models of Concurrency V*, 6900:141–159, 2012.

LRST09. Didier Lime, Olivier H. Roux, Charlotte Seidner, and Louis-Marie Traonouez. Romeo: A parametric model-checker for Petri nets with stopwatches. In Stefan Kowalewski and Anna Philippou, editors, *Proceedings of the 15th International Conference on Tools and Algorithms for the Con-*

struction and Analysis of Systems (TACAS 2009), volume 5505 of *LNCS*, pages 54–57. Springer, March 2009.

LSAF14. Giuseppe Lipari, Youcheng Sun, Étienne André, and Laurent Fribourg. Toward parametric timed interfaces for real-time components. In Étienne Andre and Goran Frehse, editors, *1st International Workshop on Synthesis of Continuous Parameters (SynCoP 2014)*, volume 145 of *Electronic Proceedings in Theoretical Computer Science*, pages 49–64, April 2014.

LSGA17. Jiaying Li, Jun Sun, Bo Gao, and Étienne André. Classification based parameter synthesis for parametric timed automata. In Zhenhua Duan and Luke Ong, editors, *Proceedings of the 19th International Conference on Formal Engineering Methods (ICFEM 2017)*, volume 10610 of *Lecture Notes in Computer Science*, pages 243–261. Springer, 2017.

NPvdP18. Hoang Gia Nguyen, Laure Petrucci, and Jaco van de Pol. Layered and collecting NDFS with subsumption for parametric timed automata. In Anthony Widjaja Lin and Jun Sun, editors, *Proceedings of the 23rd International Conference on Engineering of Complex Computer Systems (ICECCS 2018)*. IEEE, December 2018. To appear.

PGGH98. José C. Palencia Gutiérrez and Michael González Harbour. Schedulability analysis for tasks with static and dynamic offsets. In *Proceedings of the 19th IEEE Real-Time Systems Symposium (RTSS 1998)*, pages 26–37. IEEE Computer Society, 1998.

SAL15. Youcheng Sun, Étienne André, and Giuseppe Lipari. Verification of two real-time systems using parametric timed automata. In Sophie Quinton and Tullio Vardanega, editors, *Proceedings of the 6th International Workshop on Analysis Tools and Methodologies for Embedded and Real-time Systems (WATERS 2015)*, July 2015.

San15. Ocan Sankur. Symbolic quantitative robustness analysis of timed automata. In Christel Baier and Cesare Tinelli, editors, *Proceedings of the 21st International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS 2015)*, volume 9035 of *Lecture Notes in Computer Science*, pages 484–498. Springer, 2015.

SSL+13. Youcheng Sun, Romain Soulat, Giuseppe Lipari, Étienne André, and Laurent Fribourg. Parametric schedulability analysis of fixed priority real-time distributed systems. In Cyrille Artho and Peter Ölveczky, editors, *Proceedings of the Second International Workshop on Formal Techniques for Safety-Critical Systems (FTSCS 2013)*, volume 419 of *Communications in Computer and Information Science*, pages 212–228. Springer, October 2013.

TLR09. Louis-Marie Traonouez, Didier Lime, and Olivier H. Roux. Parametric model-checking of stopwatch Petri nets. *Journal of Universal Computer Science*, 15(17):3273–3304, 2009.

WTVL06. Ernesto Wandeler, Lothar Thiele, Marcel Verhoef, and Paul Lieverse. System architecture evaluation using modular performance analysis: a case study. *International Journal on Software Tools for Technology Transfer*, 8(6):649–667, 2006.