Robustness of Timed Systems

Pierre-Alain Reynier
LIF, Université d'Aix-Marseille & CNRS, France
pierre-alain.reynier@lif.univ-mrs.fr

Abstract

Formalisms used to model real-time systems include (networks of) timed automata and timed extensions of Petri nets. Timing perturbations are inherent in real-time systems, and can be due to measuring errors, imprecise clocks, non-instantaneous communications... They result in unexpected shorter or longer execution times. A real-time system robustly satisfies a property whenever this property holds in presence of small enough timing perturbations.

Above mentioned formalisms are mathematical idealizations in which these timing perturbations are completely ignored. To take them into account, we will focus on the perturbation model of guard enlargment. This model allows to ensure the existence of a correct implementation of the system, thus bridging the gap between the mathematical idealization and the finite-precision hardware. We will present several recent results, including robust model-checking and robust controller synthesis, both for timed automata and time Petri nets.

1 Motivations

Timed automata [4] and time Petri nets [17] are widely used for modelling real-time systems. These formalisms extend discrete-time models with dense-time variables, and algorithms and tools exist for model-checking these models against temporal logics.

The semantics of these models involves continuous variables, and relies on mathematical idealizations of the real-time systems. In particular, it assumes, for instance, perfect clocks for arbitrarily precise time measures, and instantaneous actions. The correctness of a model may thus depend on these unrealistic assumptions. As a consequence, given a model whose correctness has been proved w.r.t. some property, it may be impossible to build a concrete implementation which satisfies the desired property. Similarly, a synthesized controller may not be realisable on a real hardware, for instance if it should take decisions faster and faster.

In order to bridge the gap between mathematical formalisms and real implementations of them, different approaches have been proposed which study robustness of timed systems, *i.e.* their tolerance to infinitesimal timing perturbations. In this paper, we will focus on the model of guard enlargement [20], which is theoretically appealing and allows to ensure the implementability of the model. Initially considered for timed automata, we will first describe these results, and then present results obtained for other models, such as time Petri nets.

2 Timed automata

We start with a short (and incomplete) presentation of timed automata. Due to lack of space, we do not give a detailed presentation of the semantics of timed automata, and refer the reader to [5] for instance.

This work was partly supported by ANR projects ECSPER (ANR-2009-JCJC-0069) and ImpRo (ANR-2010-BLAN-0317) and by the European project Cassting (FP7-ICT-601148).

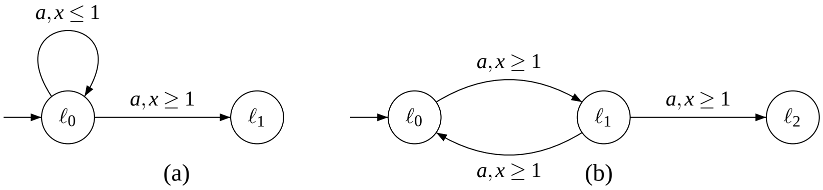

Figure 1: On the left, a timed automaton exhibiting so-called Zeno behaviours. On the right, a timed automaton from [20] in which, for any positive timing perturbations, location ℓ_2 is reachable.

A *timed automaton* $\mathscr{A}$ over some alphabet Σ is a tuple $(\mathscr{L}, \mathscr{C}, \ell_0, E)$, where $\mathscr{L}$ is a finite set of locations, $\mathscr{C}$ is a finite set of clocks, $\ell_0 \in \mathscr{L}$ is the initial location and E is a finite set of edges. Formally, an edge $e = (\ell, g, a, R, \ell') \in E$ is given by a source and a target location ℓ and ℓ', a label $a \in \Sigma$, a set of clocks R that should be reset to zero when the edge is taken, and a clock constraint g, given as a conjunction of upper and lower bounds on elements of $\mathscr{C}$.

To illustrate the robustness issues in timed automata, we consider two examples, depicted on Figure 1. On the left, the objective is to check whether or not it is possible to stay forever in the left location. Actually, considering infinite executions and as clock x is never reset, this is possible if and only if an infinitely many a's are executed within one time unit. Such behaviours, often called Zeno behaviours, are well-known unexpected behaviours of mathematical formalisms. More involved examples do exist, as one presented in [10], which has finitely many actions in any finite amount of time, but requires that the delays between two actions converge towards zero when global time diverges, which is another example of behaviour that any real device is unable to implement.

Several works have considered robustness issues for timed automata. Our aim is not to give an exhaustive list, but to survey the main directions that have been considered:

- In [14], Henzinger et al consider a topological definition whose aim was to obtain decidability of language inclusion, but this problem remains undecidable.
- The present paper studies a parametric semantics based on guard enlargement. The first work on this semantics is due to Puri [20], which studied a related parametric semantics based on drifts of clocks. More details will be given in the rest of the paper.
- Other works are related to discretization of timed automata [15, 18] or study the semantics under an unknown sampling rate [10, 1]. The goal here is to synthesize a sampling parameter under which some property holds, this is related to the implementability using digital clocks, but not directly to tolerance against imprecisions.
- An approach based on modelization is proposed in [3]. The issues of implementability are expressed directly in the model, resulting in larger models.

3 Robustness analysis using guard enlargement

Given a timed automaton $\mathscr{A}$, we denote by $\mathscr{A}_\delta$ the timed automaton obtained by enlarging its guards by the parameter δ, *i.e.* replacing every upper bound $x \leq b$ (resp. lower bound $a \leq x$) by the constraint $x \leq b + \delta$ (resp. $a - \delta \leq x$). Considering only non-negative values for δ, it is obvious that every behaviour in $\mathscr{A}$ also exists in $\mathscr{A}_\delta$.

We consider in the sequel linear-time properties which can be any ω-regular properties, or even timed properties such as those expressed in the logic MTL.

Definition 1 (Robust Model-Checking)**.** Given a linear-time property φ and a timed automaton $\mathscr{A}$, decide whether there exists $\delta_0 > 0$ such that all executions in $\mathscr{A}_{\delta_0}$ satisfy property φ. If this holds, then we say that $\mathscr{A}$ robustly satisfies φ, and that δ_0 is a witness of this satisfaction.

It is easy to observe that given two value $\delta_1 \leq \delta_2$, the set of runs of $\mathscr{A}_{\delta_1}$ is included in that of $\mathscr{A}_{\delta_2}$. As a consequence, if $\mathscr{A}$ robustly satisfies φ and if δ_0 is a witness, then, for every $\delta_0' \in [0, \delta_0]$, it holds that all the executions in $\mathscr{A}_{\delta_0'}$ satisfy property φ. This property can be understood as a "faster is better" property.

Relation to implementability The objective of robustness analysis is to guarantee the existence of a correct implementation of the model. In [13], it is shown that whenever $\mathscr{A}$ robustly satisfies some property φ, then this ensures the implementability of $\mathscr{A}$. In addition, two real characteristics of the platform of execution, namely the precision of the digital clocks and the speed of the processor, are directly related to the witness δ_0 of the robust satisfaction of $\mathscr{A}$ w.r.t. φ. The faster-is-better property states here that whenever the execution of $\mathscr{A}$ is correct with some resources (precision, speed), then it will remain correct for higher resources.

As a consequence, the following road map can be considered for the development of correct implementations of real-time systems:

1. Perform the robust model-checking of $\mathscr{A}$ against φ
2. Identify some witness δ_0
3. Implement $\mathscr{A}$, with constraints on the resources of the execution platform depending on δ_0

Note that for many kind of properties, points 1. and 2. are obtained simultaneously. We will present existing results in the next section.

4 Existing results for timed automata

Robust Model-Checking Verifying that $\mathscr{A}_{\delta_0}$ satisfies some property for some fixed δ_0 is a standard model-checking problem. The robust model-checking problem we have presented is related to parametric timed automata, which are known to be undecidable. In our context, the particular introduction of the parameter, and the monotonicity it induces in the model, allows one to preserve decidability.

We recap in the following theorem the main results known concerning the robustness analysis. We say that a timed automaton has progress cycles whenever all cycles reset each clock at least once.

Theorem 2 ([12, 6, 8, 7])**.** *Robust model-checking of safety, Büchi, LTL properties for closed timed automata is PSPACE-complete. Robust model-checking of coFlatMTL (a fragment of MTL) for closed timed automata with progress cycles is EXPSPACE-complete.*

For all those results, a witness δ_0 can be derived. The natural problem of the computation of the largest value of δ for which the result holds has been considered in [16] for the class of flat timed automata w.r.t. safety objectives.

Shrinkability Another approach has been proposed in [22]. It consists in deciding a sufficient condition for the implementability of a timed automaton $\mathscr{A}$. It can intuitively be stated as follows: is it possible to shrink the guards of $\mathscr{A}$ while preserving the behaviours of the timed automaton ?

If such a shrinking exists, then one can prove that it can be implemented in such a way that this implementation is non-blocking and preserves all time-abstract behaviours of $\mathscr{A}$.

The operation of shrinking is the dual of that of enlarging guards.

Theorem 3 ([22]). *For closed non-blocking timed automata, non-blocking-shrinkability is decidable in PSPACE.*

In addition, this approach is supported by a tool [21] called Shrinktech which is available online. Some benchmarks are given in [21].

Robust Controller Synthesis In order to model controller synthesis problems for real-time systems, a two-player game is often defined on timed automata. Controller suggests delays and actions, and the environment answers by resolving the non-determinism associated with actions, and sometimes may also choose to execute some uncontrollable action. This game formulation is well-known and has been widely studied in the "exact" framework. There has been recently important progress done to handle robustness issues in this context.

In order to lift the game formulation to the context of robustness, we will allow the environment to modify the delay proposed by the controller using some perturbation chosen in the interval $[-\delta, \delta]$. For a fixed value of δ, this defines a two player game, denoted $\mathscr{G}_\delta(\mathscr{A})$. The resulting robust game consists in deciding the existence of a positive value of δ for which controller wins the game $\mathscr{G}_\delta(\mathscr{A})$. Intuitively, the strategy of the controller should thus be tolerant to some imprecisions, which exactly corresponds to the desired property of being robust.

The game $\mathscr{G}_\delta(\mathscr{A})$ has been studied for a fixed value of δ in [11]. The parametric case has been solved for deterministic timed automata in [23] and more recently for the full class of timed automata:

Theorem 4 ([19]). *For timed automata, the robust controller synthesis is EXPTIME-complete.*

An alternative semantics has been considered in [9] which imposes less restrictions on the actions proposed by the controller. On the other hand, only reachability objectives are studied in this work.

5 Robustness of time Petri nets

Most of real-time systems are distributed by nature. Thus, in practice, systems are often modelled using networks of timed automata. An alternative formalism is that of time Petri nets. While bounded timed Petri nets can be translated into timed automata, this does not hold for general time Petri nets. In addition, the clock mechanism of time Petri nets is quite different from that of timed automata.

We have considered robustness issues in time Petri nets in [2], and studied to what extent results known for timed automata can be transferred to this model. The robust model checking of the most simple properties (preservation of the set of reachable markings) is undecidable in general. On the other hand, we have identified decidable subclasses for which different properties have a decidable robust satisfaction.

6 Perspectives

The most relevant theoretical perspectives concern robust controller synthesis. In this setting, it would be very interesting to extend our results to the presence of uncontrollable actions, and to the setting of concurrent timed games.

One can observe that the theoretical complexities of the problems presented coincide with those of the corresponding "non-robust" problems. However, while symbolic and efficient algorithms have been proposed in the "exact" setting, there is an important lack of such approaches for robust model checking and robust controller synthesis. It is thus a major challenge to make progress in this direction in order to develop the practical impact of robustness.

References

[1] Parosh Abdulla, Pavel Krčál, and Wang Yi. Sampled semantics of timed automata. *Logical Methods in Computer Science*, 6(3:14), 2010.

[2] S. Akshay, Loic Hélouet, Claude Jard, and Pierre-Alain Reynier. Robustness of time petri nets under guard enlargement. In *RP'12*, LNCS 7550, p. 92–106. Springer, 2012.

[3] Karine Altisen and Stavros Tripakis. Implementation of timed automata: An issue of semantics or modeling? In FORMATS'05, LNCS 3829, p. 273–288. Springer, 2005.

[4] Rajeev Alur and David L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2):183–235, 1994.

[5] Patricia Bouyer and Franois Laroussinie. *Modeling and Verification of Real-Time Systems*, chapter Model-Checking Timed Automata, p. 111–140. ISTE Ltd - John Wiley and Sons Ltd, 2008.

[6] Patricia Bouyer, Nicolas Markey, and Pierre-Alain Reynier. Robust model-checking of linear-time properties in timed automata. In LATIN'06, LNCS 3887, p. 238–249. Springer, 2006.

[7] Patricia Bouyer, Nicolas Markey, and Pierre-Alain Reynier. Robust analysis of timed automata via channel machines. In FoSSaCS'08, LNCS 4962, p. 157–171. Springer, 2008.

[8] Patricia Bouyer, Nicolas Markey, and Ocan Sankur. Robust model-checking of timed automata via pumping in channel machines. In FORMATS'11, LNCS 6919, p. 97–112. Springer, September 2011.

[9] Patricia Bouyer, Nicolas Markey, and Ocan Sankur. Robust reachability in timed automata: A game-based approach. In ICALP'12, LNCS 7392, p. 128–140. Springer, 2012.

[10] Franck Cassez, Thomas A. Henzinger, and Jean-François Raskin. A comparison of control problems for timed and hybrid systems. In HSCC'02, LNCS 2289, p. 134–148. Springer, 2002.

[11] Krishnendu Chatterjee, Thomas A. Henzinger, and Vinayak S. Prabhu. Timed parity games: Complexity and robustness. *Logical Methods in Computer Science*, 7(4), 2011.

[12] Martin De Wulf, Laurent Doyen, Nicolas Markey, and Jean-François Raskin. Robust safety of timed automata. *Formal Methods in System Design*, 33(1-3):45–84, 2008.

[13] Martin De Wulf, Laurent Doyen, and Jean-François Raskin. Almost ASAP semantics: From timed models to timed implementations. *Formal Aspects of Computing*, 17(3):319–341, 2005.

[14] Vineet Gupta, Thomas A. Henzinger, and Radha Jagadeesan. Robust timed automata. In HART'97, LNCS 1201, p. 331–345. Springer, 1997.

[15] T. A. Henzinger, Z. Manna, and A. Pnueli. What good are digital clocks? In *ICALP'92*, LNCS, number 623 in LNCS, p. 545–558. Springer, 1992.

[16] Rémi Jaubert and Pierre-Alain Reynier. Quantitative robustness analysis of flat timed automata. In FOSSACS'11, LNCS 6604, p. 229–244. Springer, 2011.

[17] Philip M. Merlin. *A Study of the Recoverability of Computing Systems*. PhD thesis, University of California, Irvine, CA, USA, 1974.

[18] Joël Ouaknine and James Worrell. Revisiting digitization, robustness, and decidability for timed automata. In LICS'03, p. 198–207. IEEE Comp. Soc. Press, June 2003.

[19] Youssouf Oualhadj, Pierre-Alain Reynier, and Ocan Sankur. Probabilistic robust timed games. In *CONCUR'14*, LNCS, LNCS. Springer, 2014. To appear.

[20] Anuj Puri. Dynamical properties of timed automata. *Discrete Event Dynamic Systems*, 10(1-2):87–113, 2000.

[21] Ocan Sankur. Shrinktech: A tool for the robustness analysis of timed automata. In *CAV'13*, LNCS 8044, p. 1006–1012. Springer, 2013.

[22] Ocan Sankur, Patricia Bouyer, and Nicolas Markey. Shrinking timed automata. In FSTTCS'11, LIPIcs 13, p. 375–386. Leibniz-Zentrum für Informatik, 2011.

[23] Ocan Sankur, Patricia Bouyer, Nicolas Markey, and Pierre-Alain Reynier. Robust controller synthesis in timed automata. In CONCUR'13, LNCS 8052, p. 546–560. Springer, 2013.