Pruning NDFS for Parametric Timed Automata[*]

Laure Petrucci[1] and Jaco van de Pol[2]

[1] LIPN, CNRS UMR 7030, Université Paris 13, Sorbonne Paris Cité, France
[2] Aarhus University, Denmark & University of Twente, Netherlands

Abstract. We consider parametric timed automata (PTA), in which parameters are used as bounds on clock values. We improve algorithms to synthesise the parameter constraints for which a given LTL property holds. Since this problem is undecidable, we rely on semi-algorithms that explore a possibly infinite state space of symbolic zones, represented by polyhedra. The basic synthesis algorithm applies a cumulative version of Nested Depth First Search (NDFS). Its successful termination depends crucially on heuristics for pruning the state space. We provide an overview of existing heuristic techniques, like cumulative NDFS, NDFS with subsumption, and layered NDFS. We apply these techniques to the Bounded Retransmission Protocol (BRP). We propose some optimisations based on more agressive subsumption, and show how they help solving the BRP.

1 Introduction and Motivation

Timed Automata were introduced in [1]. They have been applied to the analysis of numerous real-life timed systems, often by using the model checker Uppaal [8]. Since in reality precise timing bounds are unknown, Parametric Timed Automata (PTA) were introduced already in [2].

The simplest problem on PTAs is reachability. It comes in several variants: the emptiness variant asks if there exists a valuation of the parameters, such that an accepting location is reachable in the corresponding TA. The synthesis variant requests a description of all parameter valuations for which a certain location is reachable. Usually, model-checking tools finitely represent sets of valuations by collections of convex polyhedra. The optimisation variant [5, 15] requests some/all parameters that optimize a clock or parameter value. The reachability-emptiness problem for PTA is already undecidable. On the one hand, this led to proposals for decidable fragments, e.g. L/U-PTAs. On the other hand, this led to the development of semi-algorithms, which can either diverge or provide a correct answer. Well known algorithms are based on the inverse method, or on the exploration of the parametric zone graph (PZG), whose zones are represented by convex polyhedra. Although the PZG is infinite in general, techniques like subsumption, state merging [6] and smart exploration orders [7] have been introduced to reduce the state space, and increase the chance of termination of the algorithm. Many of these algorithms are implemented in Imitator [3].

[*] This research was conducted with the support of PHC Van Gogh project PAMPAS.

Here, we are interested in the Linear Time Logic (LTL) analysis and LTL synthesis problems. LTL admits the specification of safety and liveness properties for reactive systems. In the automata-based approach, an LTL formula ϕ is transformed into a Büchi automaton $B_{\neg\phi}$, so checking if $M \models \phi$ reduces to checking that there is no accepting cycle in $M \times B_{\neg\phi}$. In our case, M is a PTA. The basic procedure would be to apply the Nested Depth-First Search. A blue search identifies accepting states in post-order, launching a "nested" red search to detect accepting cycles, starting from the accepting states as seed. The NDFS algorithm runs on the PZG of the product system. Here each symbolic state consists of two parts: (1) the locations and discrete data values of each automaton in the network including the property automaton, and (2) a linear constraint on clock and parameters values as a convex polyhedron, called a zone.

The main problem is that the basic algorithm suffers from non-termination, already in simple cases. For reachability, this is usually solved by subsuming states whose zone is included already in the zone of other states with the same locations. It is well known that the breadth-first exploration strategy tends to find states with a large zone first, thus effectively pruning large parts of the state space. However, NDFS for checking LTL properties is inherently depth-first, so it can easily diverge in an infinite branch. Even worse, applying inclusion abstraction (based on subsumption) is not sound, since the abstraction can introduce spurious accepting cycles that are not present in the concrete state space.

This papers explores a number of existing approaches to prune the state space, and increase the chance for termination for LTL model checking. We applied them to some case studies and more specifically to the Bounded Retransmission Protocol, and suggest some further improvements.

2 Different Pruning Strategies

The liveness problem for timed automata is inherently more difficult than the reachability problem, as shown in [13]. Still, we are interested in heuristics that can solve realistic problems, for instance parameter synthesis for LTL properties for the Bounded Retransmission Protocol.

We will summarize four heuristics (A-D) that are based on the following three observations on the PZG: (1) In every transition in the PZG, the projection of the constraints on the parameters is (weakly) decreasing. (2) Consequently, all states on an (accepting) cycle have the same parametric zone. (3) Any path from an accepting state to a larger state (spiral) can be extended to an accepting cycle.

The basic approach to LTL synthesis is *cumulative NDFS (A)* [9, 16]. It runs a standard NDFS, but when an accepting cycle is detected, the search is not aborted: it continues after collecting the current parameter constraint in a global disjunction of parameter constraints leading to a property violation. This also introduces an immediate opportunity for pruning: if we encounter a symbolic state whose parametric zone is included in the global one, then exploring this one will not contribute to the global constraint (according to obeservation 1), so we can immediately backtrack.

It has been shown that NDFS for timed automata can be combined with *subsumption (B)* [14]. This requires special care, since the full abstraction may introduce spurious accepting cycles. First, we can discard states that are subsumed by states that have already been fully explored by the nested "red" search, (i.e. included in a "red state"). On the other hand, a cycle is reported or collected as soon as a state is encountered that includes the seed state (following observation3). We recently extended this result to parametric timed automata [16].

In that paper [16], we also introduced a *layering (C)* approach, based on observation 2: If during the red search for a cycle a transition is traversed where the projection on the parameters strictly decreases, this transition cannot be part of the cycle. Actually, all states with a smaller projected parameter constraint can moved to a "pending list" for later treatment. This partly restores the good tendency of the breadth-first strategy, to treat large symbolic states first.

Finally, [13] proposes to first analyse the *overapproximation (D)* obtained by full inclusion abstraction. If there is no cycle in this abstract state space, we are done. However, if we encounter a cycle in the abstract state space, we must still check if it is spurious. To this end, they run an analysis without subsumption in the current strongly connected component.

3 Synthesis for the Bounded Retransmission Protocol

The Bounded Retransmission Protocol (BRP) was initially used by Philips in remote TV controllers using infrared light. A time-abstract version was formally verified correct in [12], and a parameterised timed version was studied in [10]. A particular version of BRP is included in the IMITATOR benchmark set [4]. Our study here is focussed on a cleaned up version of this benchmark specification. In addition we added some LTL properties. We used Spot [11] to generate a Bchi automaton for the properties, and manually added them to the IMITATOR model as an extra monitor component, thus introducing accepting states. Examples of properties include:

- $\mathbf{G}\,\neg R_{bad}$: The receiver never gets an unexpected frame (this depends on the timing parameters; we synthesised a slightly more liberal constraint than [10])
- $\mathbf{G}\,\mathbf{F}\,S_{in}$: The protocol is able to handle an infinite amount of requests (this holds since the BRP is *bounded*)

We noticed that initially, cumulative NDFS with subsumption and/or layering could not synthesize these constraints: subsumption only helps *after* detecting some accepting cycles. Without subsumption, some clocks kept growing, leading to exploring smaller and smaller constraints.

We improved the algorithm for LTL analysis by including early cycle detection in the blue search: if a transition to an accepting state on the stack is found, there is no need to call the red search, but the current constraint can be directly collected. Also ordering the successors, to treat accepting states before non-accepting states improved the algorithm. Finally, to improve the layered approach, we propose to apply the overapproximation method from [13] per layer. Initial manual experiments showed that we could reconstruct some constraints in this combined approach that lead to a violation of $\mathbf{G}\,\neg R_{bad}$.

We revisited several case studies from [16] that diverged both with subsumption and layering. In three more cases (critical-region, F4 and JLR13) we can now at least find some constraints that violate the property. In critical-section, we even terminate with three symbolic states, where the original algorithms all diverged. In JLR13, an accepting cycle was found with two states only.

References

1. R. Alur and D. L. Dill. The theory of timed automata. In *REX Workshop*, LNCS 600, pages 45–73. Springer, 1991.
2. R. Alur, T. A. Henzinger, and M. Y. Vardi. Parametric real-time reasoning. In *STOC*, pages 592–601. ACM, 1993.
3. É. André. IMITATOR: A tool for synthesizing constraints on timing bounds of timed automata. In *ICTAC*, LNCS 5684, pages 336–342. Springer, 2009.
4. É. André. A benchmark library for parametric timed model checking. In *FTSCS*, volume 1008 of *CCIS*, pages 75–83. Springer, 2018.
5. É. André, V. Bloemen, L. Petrucci, and J. van de Pol. Minimal-time synthesis for parametric timed automata. In *TACAS*, LNCS. Springer, 2019.
6. É. André, L. Fribourg, and R. Soulat. Merge and conquer: State merging in parametric timed automata. In *ATVA*, LNCS 8172, pages 381–396. Springer, 2013.
7. É. André, H. G. Nguyen, and L. Petrucci. Efficient parameter synthesis using optimized state exploration strategies. In *ICECCS*, pages 1–10. IEEE Computer Society, 2017.
8. G. Behrmann, A. David, K. G. Larsen, J. Håkansson, P. Pettersson, W. Yi, and M. Hendriks. UPPAAL 4.0. In *QEST*. IEEE Computer Society, 2006.
9. P. Bezdek, N. Benes, J. Barnat, and I. Cerná. LTL parameter synthesis of parametric timed automata. In *SEFM*, LNCS 9763, pages 172–187. Springer, 2016.
10. P. R. D'Argenio, J. Katoen, T. C. Ruys, and J. Tretmans. The bounded retransmission protocol must be on time! In *TACAS*, LNCS 1217. Springer, 1997.
11. A. Duret-Lutz, A. Lewkowicz, A. Fauchille, T. Michaud, E. Renault, and L. Xu. Spot 2.0 - A framework for LTL and ω -automata manipulation. In *ATVA*, volume 9938 of *Lecture Notes in Computer Science*, pages 122–129, 2016.
12. J. F. Groote and J. van de Pol. A bounded retransmission protocol for large data packets. In *AMAST*, LNCS 1101, pages 536–550. Springer, 1996.
13. F. Herbreteau, B. Srivathsan, T. Tran, and I. Walukiewicz. Why liveness for timed automata is hard, and what we can do about it. In *FSTTCS*, LIPIcs 65, pages 48:1–48:14. Schloss Dagstuhl - Leibniz-Z. fr Informatik, 2016.
14. A. Laarman, M. C. Olesen, A. E. Dalsgaard, K. G. Larsen, and J. van de Pol. Multi-core emptiness checking of timed Büchi automata using inclusion abstraction. In *CAV*, LNCS 8044, pages 968–983. Springer, 2013.
15. L. Luthmann, T. Gerecht, A. Stephan, J. Bürdek, and M. Lochau. Minimum/maximum delay testing of product lines with unbounded parametric real-time constraints. *Journal of Systems and Software*, 149:535–553, 2019.
16. H. G. Nguyen, L. Petrucci, and J. van de Pol. Layered and collecting NDFS with subsumption for parametric timed automata. In *ICECCS*, pages 1–9. IEEE, 2018.