Automated Verification for Real-Time Systems using Implicit Clocks and an Extended Antimirov Algorithm

Yahui Song
National University of Singapore
Singapore, Singapore
yahuis@comp.nus.edu.sg

Wei-Ngan Chin
National University of Singapore
Singapore, Singapore
chinwn@comp.nus.edu.sg

Abstract

The correctness of real-time systems depends both on the correct functionalities and the realtime constraints. To go beyond the existing Timed Automata based techniques, we propose a novel solution that integrates a modular Hoare-style forward verifier with a new term rewriting system (TRS) on *Timed Effects* (*TimEffs*).

The main purposes are to increase the expressiveness, dynamically create clocks, and efficiently solve constraints on the clocks. We formally define a core language $\mathtt{C}^{\mathtt{t}}$, generalizing the real-time systems, modeled using mutable variables and timed behavioral patterns, such as *delay*, *deadline*, *interrupt*, etc. Secondly, to capture real-time specifications, we introduce *TimEffs*, a new effects logic, that extends *Regular Expressions* with dependent values and arithmetic constraints. Thirdly, the forward verifier infers temporal behaviors of given $\mathtt{C}^{\mathtt{t}}$ programs, expressed in *TimEffs*. Lastly, we present a purely algebraic term rewriting system, to efficiently prove language inclusions between *TimEffs*. To demonstrate the proposal's feasibility, we prototype the verification system; prove its soundness; report on experimental results.

CCS Concepts: • **Theory of computation** → **Logic and verification**; **Program verification**; *Automated reasoning*; Linear logic.

Keywords: Temporal Verification, Dependant Effects, Term Rewriting System, Timed Verification

ACM Reference Format:
Yahui Song and Wei-Ngan Chin. 2022. Automated Verification for Real-Time Systems using Implicit Clocks and an Extended Antimirov Algorithm. In *Companion Proceedings of the 2022 ACM SIGPLAN International Conference on Systems, Programming, Languages, and Applications: Software for Humanity (SPLASH Companion '22), December 5–10, 2022, Auckland, New Zealand*. ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3563768.3563953

SPLASH Companion '22, December 5–10, 2022, Auckland, New Zealand

ACM ISBN 978-1-4503-9901-2/22/12.
https://doi.org/10.1145/3563768.3563953

1 Introduction

Specification and verification of real-time systems are essential to research topics with practical implications. During the last more than two decades, a popular approach for specifying real-time systems has been based on Timed Automata [1]. Timed Automata are powerful in designing real-time models with explicit clock variables. Real-time constraints are captured by explicitly setting/resetting clock variables. A number of automatic verification support for Timed Automata have proven to be successful [4, 8–10].

Models based on Timed Automata often adopt a simple structure, e.g., a network with no hierarchy. The benefit is that efficient model checking is made feasible. Nonetheless, designing and verifying compositional real-time systems is becoming an increasingly difficult task due to the widespread applications and increasing complexity of such systems. Unlike timed process algebras, Timed Automata lack high-level compositional patterns for hierarchical design. As a result, users often need to manually cast those terms into a set of clock variables with carefully calculated clock constraints. The process is tedious and error-prone.

We investigate an alternative approach for modeling and verifying compositional real-time systems. In this work, we propose a novel temporal specification language, which enables a compositional verification via a Hoare-style forward verifier and a term rewriting system (TRS). More specifically, we specify system behaviors in the form of Timed Effects (*TimEffs*), which integrates the Kleene Algebra with dependent values and arithmetic constraints, to provide real-time abstractions into traditional linear temporal logics. For example, one safety property, *"The event* ***Done*** *will be triggered no later than one time unit"*[1], is expressed in *TimEffs* as:

$$\Phi \triangleq \mathtt{0{\leq}t{<}1} \wedge (\_^{\star} \cdot \textbf{Done})\#\mathtt{t}$$

Here, ∧ connects the arithmetic formula and the timed trace, # is a novel operator specifying the *real-time* constraints for the *logical-time* sequences [7]; _ is a wildcard matching to any event; Kleene star ★ denotes trace repetition. Moreover,

[1] Without loss of generality, we use integer values to represent time units in this paper.

the time bounds can be dependent on the program inputs, demonstrated in Figure 1., where preconditions and postconditions are marked by *req* and *ens*. Function `addNSugar` takes a parameter `n`, representing the portion of the sugar we need to add. When `n=0`, it simply raises an event **EndSugar** to mark the end of the process. Otherwise, it adds one portion of the sugar by calling `addOneSugar()`, then recursively calls `addNSugar` with parameter `n-1`. The use of statement `timeout(e, d)` is standard [5], which executes a block of code `e` after the specified time `d`. Therefore, the time spent on adding one portion of the sugar is more than one time unit. Note that $\epsilon\#t$ refers to an empty trace which takes time `t`. Both preconditions require no arithmetic constraints, and have no temporal constraints upon the history traces. The postcondition of `addNSugar(n)` indicates that the method generates a finite trace where **EndSugar** takes a no less than `n` time-units delay to finish.

```
void addOneSugar()
/* req: true ∧ _*
   ens: t>1 ∧ ϵ # t */
{ timeout ((), 1); }

void addNSugar(int n)
/* req: true ∧ _*
   ens: t≥n ∧ EndSugar#t*/
{ if (n == 0)
     event["EndSugar"]
   else {
     addOneSugar();
     addNSugar(n-1);}}
```

Figure 1. Value dependent specification.

Although these examples are simple, they show the benefits of deploying value-dependent time bounds. Intuitively, if traditional Timed Automata define an *exact* transition system, *TimEffs* define a set of exact transition systems.

Moreover, we deploy a Hoare-style forward verifier to soundly infer the actual behaviors of given programs concerning the well-defined operational semantics. This approach provides a *direct* (opposite to the techniques which require manual and remote modeling processes), and modular compositional verification for real-time systems, which are not possible by existing techniques.

Having *TimEffs* to be the specification language, and the forward verifier to infer the program behaviors, we are interested in the following verification problem: Given a program $\mathcal{P}$, and a temporal specification Φ', does the inclusions $\Phi^{\mathcal{P}} \sqsubseteq \Phi'$ holds? Typically, checking the inclusion/entailment between the concrete program effects $\Phi^{\mathcal{P}}$ and the valid traces Φ' proves that: the program $\mathcal{P}$ will never lead to unsafe traces which violate Φ'. The expressiveness of *TimEffs* goes beyond finite-state automata, it is not possible to translate them into Timed Automata and rely on the solving engines of Timed Automata. Therefore we develop a novel TRS, which is inspired by Antimirov and Mosses' algorithm[2] [2] but solving the language inclusions between more expressive *TimEffs*.

2 Verification Overview

The proposed verification framework is shown in Figure 2. Rounded boxes are the main procedures, and both return *true* when the forward reasoning/proving succeeds, return *false* otherwise. Rectangular boxes describe the inputs to the procedures. The forward verifier relies on the TRS.

The inputs of the forward verifier are target programs annotated with temporal specifications. The input of the TRS is a pair of effects LHS and RHS, referring to the inclusion LHS ⊑ RHS to be checked *(⊑ captures the inclusion relation between effects. LHS refers to left-hand-side effects, and RHS refers to right-hand-side effects.)*. Besides, the verifier calls the TRS to prove produced inclusions, i.e., between the current effects states and assertions. Our main contributions are:

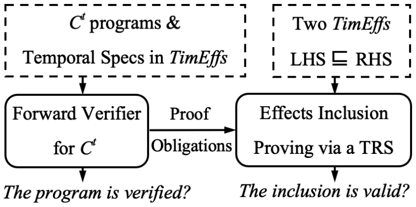

Figure 2. System Overview.

1. **Language Abstraction:** we define a core language $\mathtt{C}^t$, via its syntax and semantics, generalizing the real-time systems with mutable variables and timed behavioral patterns.
2. **Specification Language:** we propose *TimEffs*, by defining its syntax and semantics, gaining the expressive power beyond traditional modeling languages for real-time systems.
3. **Automated Forward Verifier:** we establish a sound axiomatic semantics to infer the temporal behaviors of given target programs. The verifier triggers the back-end TRS.
4. **An Efficient TRS:** we present the rewriting rules to prove the inclusion relations between the inferred behaviors and the given temporal specifications, both in *TimEffs*.
5. **Implementation and Evaluation:** we prototype *TimEffs* and the automated verification system, prove the soundness, report on case studies and experimental results.

3 Implementation and Evaluation

We prototype our automated verification system using OCaml. The proof obligations, for arithmetic constraints, generated by the verifier are discharged using constraint solver Z3 [3]. We prove termination and soundness of both the forward verifier and the TRS. We validate our implementation against the state-of-the-art PAT [6] model checker for conformance.

4 Conclusion

We define the syntax and semantics of *TimEffs*, to capture real-time systems' behaviors and temporal properties. We demonstrate how to give an axiomatic semantics to $\mathtt{C}^t$ by timed-trace processing functions, which enables our Hoare-style forward verifier, to constructively compute the program effects. We present an effects inclusion checker (the TRS) to prove the annotated temporal properties efficiently. We prototype the verification system and show its feasibility.

[2] Antimirov and Mosses' algorithm was designed for deciding the inequalities of regular expressions based on an axiomatic algorithm of the algebra of regular sets.

References

[1] Rajeev Alur and David L. Dill. 1994. A Theory of Timed Automata. *Theor. Comput. Sci.* 126, 2 (1994), 183–235. https://doi.org/10.1016/0304-3975(94)90010-8

[2] Valentin M. Antimirov and Peter D. Mosses. 1995. Rewriting Extended Regular Expressions. *Theor. Comput. Sci.* 143, 1 (1995), 51–72. https://doi.org/10.1016/0304-3975(95)80024-4

[3] Leonardo Mendonça de Moura and Nikolaj Bjørner. 2008. Z3: An Efficient SMT Solver. In *Tools and Algorithms for the Construction and Analysis of Systems, 14th International Conference, TACAS 2008, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2008, Budapest, Hungary, March 29-April 6, 2008. Proceedings (Lecture Notes in Computer Science)*, C. R. Ramakrishnan and Jakob Rehof (Eds.), Vol. 4963. Springer, 337–340. https://doi.org/10.1007/978-3-540-78800-3_24

[4] Kim Guldstrand Larsen, Paul Pettersson, and Wang Yi. 1997. UPPAAL in a Nutshell. *Int. J. Softw. Tools Technol. Transf.* 1, 1-2 (1997), 134–152. https://doi.org/10.1007/s100090050010

[5] Parewa Labs Pvt. Ltd. 2022. https://www.programiz.com/javascript/setTimeout.

[6] Jun Sun, Yang Liu, Jin Song Dong, and Jun Pang. 2009. PAT: Towards Flexible Verification under Fairness. In *Computer Aided Verification, 21st International Conference, CAV 2009, Grenoble, France, June 26 - July 2, 2009. Proceedings (Lecture Notes in Computer Science)*, Ahmed Bouajjani and Oded Maler (Eds.), Vol. 5643. Springer, 709–714. https://doi.org/10.1007/978-3-642-02658-4_59

[7] Reinhard von Hanxleden, Timothy Bourke, and Alain Girault. 2017. Real-time ticks for synchronous programming. In *2017 Forum on Specification and Design Languages, FDL 2017, Verona, Italy, September 18-20, 2017*, Franco Fummi, Hiren D. Patel, and Samarjit Chakraborty (Eds.). IEEE, 1–8. https://doi.org/10.1109/FDL.2017.8303893

[8] Farn Wang, Rong-Shiung Wu, and Geng-Dian Huang. 2005. Verifying Timed and Linear Hybrid Rule-Systems with RED. In *Proceedings of the 17th International Conference on Software Engineering and Knowledge Engineering (SEKE’2005), Taipei, Taiwan, Republic of China, July 14-16, 2005*, William C. Chu, Natalia Juristo Juzgado, and W. Eric Wong (Eds.). 448–454.

[9] Xinyu Wang, Jun Sun, Ting Wang, and Shengchao Qin. 2017. Language Inclusion Checking of Timed Automata with Non-Zenoness. *IEEE Trans. Software Eng.* 43, 11 (2017), 995–1008. https://doi.org/10.1109/TSE.2017.2653778

[10] Sergio Yovine. 1997. KRONOS: A Verification Tool for Real-Time Systems. *Int. J. Softw. Tools Technol. Transf.* 1, 1-2 (1997), 123–133. https://doi.org/10.1007/s100090050009