Asynchronous composition of temporal logics

Alberto Bombardelli[1,2][0000−0003−3385−3205]

[1] Fondazione Bruno Kessler, Via Sommarive, 18, 38123 Povo TN
{abombardelli}@fbk.eu
[2] University of Trento, Via Sommarive, 9, 38123 Povo TN
{alberto.bombardell-1}@unitn.it

Abstract. Compositional reasoning permits the application of a divide-and-conquer approach by decomposing the verification problem into many sub-problems. Instead of verifying the full- scale system, it is possible to verify local properties in isolation, and then compose them to entail the global property. This technique appears to be effective when dealing with synchronous systems, where the composition of components is always simple and well- defined. However, when dealing with asynchronous systems, the problem complicates due to the non-deterministic interleaving of components, concurrent access to shared variables, synchronizations and timing constraints. Our study investigates compositional approaches to deal with linear temporal logics, MTL with both local and global time semantics in both discrete and timed domains. Moreover, we integrate our approach into COMPASTA, a GUI tool for the verification and validation of asynchronous systems.

1 Introduction

Asynchronous systems are systems in which different components can run concurrently. These systems are ubiquitous in several domains such as the automotive domain, aerospace domain and distributed system domain. Bugs in these systems are often difficult to spot and can cause severe consequences. Although testing[6] and simulation[7] have proven helpful in spotting bugs and incorrect specifications in these systems, these techniques are incomplete; therefore, they can not guarantee the correctness of a system. Model checking[11] is an algorithmic approach used to automatically verify the correctness of models. SMT-based model checking is a prominent technique to verify infinite-state systems. Usually, in model checking, properties are expressed as formal temporal logic specifications such as LTL[17]. When dealing with real-time systems and their properties, one of the most popular temporal logic is Metric Temporal Logic (MTL) [14], which enriches the temporal operators with bounds to constrain the time intervals in which formulas must be satisfied. One of the issues to specify and reason with MTL properties in Distributed Real-Time System is that clocks are not perfectly synchronized and the nodes of a distributed system may refer to different, possibly skewed, clocks. Distributed variants of MTL and ECTL use local temporal operators that refer to local times (e.g., [15]), which are independent but usually strictly increasing.

Compositional reasoning[18] is a divide-and-conquer approach based on the correct-by-construction principle. This approach has been proven effective in tackling the state explosion problem that affects model checking. However, systems with complex interactions between components pose a threat to such improvements because they make compositional approaches harder to apply and less effective.

In this study, we propose a compositional methodology to verify the correctness of complex asynchronous systems with complex interactions. We focus on the asynchronous composition of LTL and MTL with both global and local time semantics and we integrate our techniques into the validation & verification tool COMPASTA.

2 Contribution

2.1 Asynchronous composition of Interface LTL properties

In [2], we defined our approach for the compositional verification of first-order LTL with event freezing functions[9,19] properties. We consider the composition of properties where components share data port variables. Moreover, with this asynchronous model, it is possible to represent more complex style of communication such as communication buffers and asynchronous messages. This approach is based on the notion of LTL properties composition, where the asynchronous composition of respectively LTL and the components are defined by the γ_P and γ_S functions of the following inference rule[18]:

Inference 1 *Let $\mathcal{M}_1, \ldots, \mathcal{M}_n$ be a set of n components, $\varphi_1, \ldots, \varphi_n$ be local properties on each component, The following inference is true:*

$$\frac{\dfrac{\mathcal{M}_1 \models \varphi_1, \ldots, \mathcal{M}_n \models \varphi_n}{\gamma_S(\mathcal{M}_1, \ldots, \mathcal{M}_n) \models \gamma_P(\varphi_1, \ldots, \varphi_n)} \quad \gamma_P(\varphi_1, \ldots, \varphi_n) \models \varphi}{\gamma_S(\mathcal{M}_1, \ldots, \mathcal{M}_n) \models \varphi}$$

First, we formalized a symbolic notion of the asynchronous composition of transition systems sharing variables to I/O data ports. From this notion of transition system composition, we derive a formal definition of projection and its inverse. In this work, the projection restricts the trace symbols to the local ones and removes the inactive states called stuttering states. From the notion of projection and stuttering, we produce a rewriting-based approach for the composition of the LTL local properties. We defined the rewriting transformation of the properties embedding the stuttering variable in various forms depending on the semantics of the formula. We use the inverse of projection to demonstrate that each global trace satisfies the rewriting of the local property if and only if the local trace satisfies the local property. Then, we take advantage of the structure of the problem to produce an optimization to mitigate the blow-up in terms of formula size provided by the non-optimized rewriting. Finally, we produce an alternative technique that translates the local properties into an automaton and then apply the asynchronous composition defined for γ_S. The approaches have been implemented in the state-of-the-art contract-based design tool OCRA[8] and we carried out an experimental evaluation on Dwyer pattern LTL formulae[12]. Currently, we are extending this approach to support scenarios in which local components run a finite amount of time. To do so, we are adapting our compositional framework to support the truncated semantics of LTL[13] extended with predicates, functions and past operators. Finally, this approach has been successfully employed in contract-based design by extending the contract proof system of [10] with our rewriting. In the future, we would like to consider the synthesis problem applied to this setting. In particular, it would be interesting to synthesis top level contracts from component specifications and scheduling constraints; similarly, synthesis could be applied to find proper scheduling constraints that guarantee a correct composition.

2.2 MTLSK composition

In the context of distributed real-time systems, local clocks are not perfectly synchronized with global time. It is often the case that such systems synchronize their clocks through local resets. Distributed variants of MTL and ECTL use local temporal operators that refer to local times (e.g., [15]), which are independent but usually strictly increasing. In [3,4], we considered a distributed extension of MTL and its verification. The semantics of MTL operators were evaluated on skewed clocks that get synchronized. Currently, we are considering extension of this work applying compositional verification as for [2]; moreover, we would like to validate our approach with a robust case study.

2.3 COMPASTA

Our last contribution is in the context of the COMPASTA project. COMPASTA[1] is a project funded by the European Space Agency (ESA) to combine the TASTE[16] design tool with the verification capabilities of COMPASS[5] tool in a unique modelling tool. This tool permits the application of extensive verification and validation capabilities to models representing software and hardware system that interacts asynchronously. The verification capabilities that this tool offers are requirements analysis, contract-based design, functional verification and safety assessment, fault detection and identification analysis. In this tool, we integrated our compositional approach inside contract-based design analysis and we defined the semantics of the component interactions taking into account the presence of a scheduler and the fact that the system is composed of both software and hardware components.

References

1. A. Bombardelli, M. Bozzano, R. Cavada, A. Cimatti, A. Griggio, M. Nazaria, E. Nicolodi, and S. Tonetta. Compasta: Extending taste with formal design and verification functionality. In *Model-Based Safety and Assessment: 8th International Symposium, IMBSA 2022, Munich, Germany, September 5–7, 2022, Proceedings*, page 21–27, Berlin, Heidelberg, 2022. Springer-Verlag.

2. A. Bombardelli and S. Tonetta. Asynchronous Composition of Local Interface LTL Properties. In *NFM*, pages 508–526, 2022.
3. A. Bombardelli and S. Tonetta. Metric temporal logic with resettable skewed clocks. In *2023 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2023.
4. A. Bombardelli and S. Tonetta. Reasoning with metric temporal logic and resettable skewed clocks. In *NASA Formal Methods: 15th International Symposium, NFM 2023, Houston, TX, USA, May 16–18, 2023, Proceedings*, page 174–190, Berlin, Heidelberg, 2023. Springer-Verlag.
5. M. Bozzano, H. Bruintjes, A. Cimatti, J.-P. Katoen, T. Noll, and S. Tonetta. Compass 3.0. In T. Vojnar and L. Zhang, editors, *Tools and Algorithms for the Construction and Analysis of Systems*, pages 379–385, Cham, 2019. Springer International Publishing.
6. M. Broy, B. Jonsson, J. Katoen, M. Leucker, and A. Pretschner, editors. *Model-Based Testing of Reactive Systems, Advanced Lectures*, volume 3472 of *Lecture Notes in Computer Science*. Springer, 2005.
7. F. Cellier and E. Kofman. Continuous system simulation. *Continuous System Simulation*, pages 1–643, 01 2006.
8. A. Cimatti, M. Dorigatti, and S. Tonetta. Ocra: A tool for checking the refinement of temporal contracts. pages 702–705, 11 2013.
9. A. Cimatti, A. Griggio, E. Magnago, M. Roveri, and S. Tonetta. Smt-based satisfiability of first-order ltl with event freezing functions and metric operators. *Information and Computation*, 272:104502, 12 2019.
10. A. Cimatti and S. Tonetta. Contracts-refinement proof system for component-based embedded systems. *Science of Computer Programming*, 97:333–348, 2015. Object-Oriented Programming and Systems (OOPS 2010) Modeling and Analysis of Compositional Software (papers from EUROMICRO SEAA'12).
11. E. Clarke, T. Henzinger, H. Veith, and R. Bloem. *Handbook of Model Checking*. Springer International Publishing, 2018.
12. M. Dwyer, G. Avrunin, and J. Corbett. Patterns in property specifications for finite-state verification. *Proceedings - International Conference on Software Engineering*, 02 1970.
13. D. Fisman and H. Kugler. Temporal reasoning on incomplete paths. In T. Margaria and B. Steffen, editors, *Leveraging Applications of Formal Methods, Verification and Validation. Verification*, pages 28–52, Cham, 2018. Springer International Publishing.
14. R. Koymans. Specifying real-time properties with metric temporal logic. *Real-Time Syst.*, 2(4):255–299, oct 1990.
15. J. J. Ortiz, A. Legay, and P. Schobbens. Distributed Event Clock Automata - Extended Abstract. In *CIAA*, pages 250–263, 2011.
16. M. Perrotin, E. Conquet, J. Delange, A. Schiele, and T. Tsiodras. Taste: A real-time software engineering tool-chain overview, status, and future. In I. Ober and I. Ober, editors, *SDL 2011: Integrating System and Software Modeling*, pages 26–37, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.
17. A. Pnueli. The temporal logic of programs. pages 46–57, 09 1977.
18. W.-P. Roever, F. Boer, U. Hannemann, J. Hooman, Y. Lakhnech, M. Poel, and J. Zwiers. *Concurrency Verification: Introduction to Compositional and Noncompositional Methods.* 01 2001.
19. S. Tonetta. Linear-time temporal logic with event freezing functions. *Electronic Proceedings in Theoretical Computer Science*, 256, 09 2017.