8-2012

Automatic compositional verification of timed systems

Shang-Wei LIN

Yang LIU

Jun SUN
Singapore Management University, junsun@smu.edu.sg

Jin Song DONG

Étienne ANDRÉ

Automatic Compositional Verification of Timed Systems

Shang-Wei Lin[1], Yang Liu[1], Jun Sun[2], Jin Song Dong[3], and Étienne André[4]

[1] Temasek Laboratories, National University of Singapore*
{tsllsw,tslliuya}@nus.edu.sg
[2] Singapore University of Technology and Design
sunjun@sutd.edu.sg
[3] National University of Singapore
dongjs@comp.nus.edu.sg
[4] LIPN, CNRS UMR 7030, Université Paris 13, France
Etienne.Andre@lipn.univ-paris13.fr

Abstract. Specification and verification of real-time systems are important research topics with crucial applications; however, the so-called state space explosion problem often prevents model checking to be used in practice for large systems. In this work, we present a self-contained toolkit to analyze real-time systems specified using *event-recording automata* (ERAs), which supports system modeling, animated simulation, and fully automatic compositional verification based on learning techniques. Experimental results show that our tool outperforms the state-of-the-art timed model checker.

1 Introduction

Ensuring the correctness of safety-critical systems with timing requirements is crucial and challenging. Model checking is emerging as an effective verification method and has been widely used for timed system. However, model checking suffers from the infamous state space explosion problem, and the problem is even graver in timed model checking because of the timed transitions.

To alleviate this problem, we proposed an automatic learning-based compositional verification framework for timed systems (cf. technical repoert [7]). We focus on timed systems that are modeled by *event-recording automata* (ERAs) [1], which is a determinizable class of timed automata. ERAs are as powerful as timed transition systems and are sufficiently expressive to model many interesting timed systems. The proposed framework consists of a compositional verification based on the non-circular assume-guarantee (AG-NC) proof rule [9] and uses a learning algorithm, TL* [8], to automatically generate timed assumptions for *assume-guarantee reasoning* (AGR).

Our engineering efforts realize the proposed techniques into a self-contained toolkit for analyzing real-time systems, which is built as the ERA module (can be downloaded

* This work is mainly supported by TRF Project "Research and Development in the Formal Verification of System Design and Implementation" from Temasek Lab@National University of Singapore; partially supported by project IDG31100105/IDD11100102 from Singapore University of Technology and Design, project MOE2009-T2-1-072 from School of Computing@National University of Singapore, and Merlion Project R-252-000-482-133.

D. Giannakopoulou and D. Méry (Eds.): FM 2012, LNCS 7436, pp. 272–276, 2012.

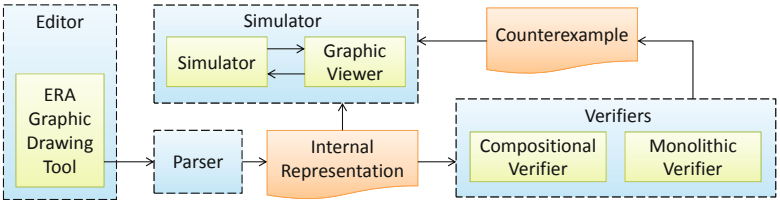
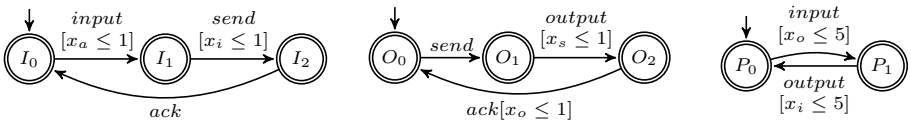


Fig. 1. Architecture of the ERA Module in PAT



(a) INPUT (b) OUTPUT (c) Property φ

Fig. 2. Models and property of the I/O system

at [6]) in the PAT model checker [10]. Fig. 1 shows the architecture of our tool, which consists of four components, namely the *editor*, the *parser*, the *simulator* and *verifiers*. The editor is featured with a powerful graphic drawing component that allows users to design system models and specify properties by drawing ERAs. The editor also supports syntax highlighting, intellisense, and undo/redo functionality such that designers can efficiently model the systems. The parser compiles both the system models and the properties (in the form of ERAs) into internal representations for simulation and verification. The simulator allows users to perform various simulation tasks on the input model such as user interactive simulation, trace replay and so on. Most importantly, compositional verification is fully automated for *safety* properties specified using ERAs. To the best of our knowledge, our tool is the first one supporting fully automatic compositional verification for timed systems. Our tool also supports the traditional monolithic approach that generates the global state space based on zone abstraction. Users can choose to use either the monolithic or our compositional approach inside the verification interface. If the verification result is false, counterexamples will be produced and can be visualized using the simulator. Experimental results (Section 3) show that our tool of compositional verification for real-time systems outperforms traditional timed monolithic approaches in many cases.

2 Compositional Verification of ERAs

An *event-recording automaton* (ERA) is a special case of timed automaton where each event a on a transition is associated with a corresponding event-recording clock x_a recording the time elapsed since the last occurrence of event a. Each event-recording clock x_a is implicitly and automatically reset when a transition with event a is taken.

Fig. 2 gives an I/O system with two components, INPUT and OUTPUT, modeled by ERAs. The pairs of event-recording clocks and the corresponding events are $x_i : input$, $x_s : send$, $x_o : output$, and $x_a : ack$. The model of the INPUT component is

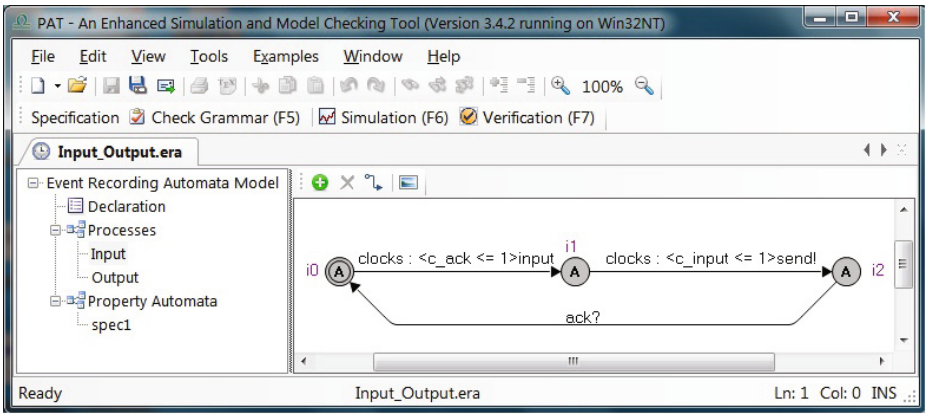


Fig. 3. GUI of the PAT Model Checker

shown in Fig. 2 (a). It performs an $input$ event within one time unit once it receives an ack event from OUTPUT. Subsequently, it performs a $send$ event to notify OUTPUT and waits for another ack event from OUTPUT. The model of OUTPUT is shown in Fig. 2 (b), which is similar to INPUT. The system property φ, as shown in Fig. 2 (c), is that $input$ and $output$ events should alternate and the time difference between every two consecutive events should not exceed five time units. Fig. 3 shows the INPUT component modeled in PAT, where a double circle represents the initial state and a state labeled with "**A**" represents an accepting state.

The flow of the proposed timed compositional verification is a two-phase process using the TL* algorithm [8] to automatically learn the timed assumption needed by AGR. The first, *untimed verification*, phase constructs the untimed assumption, and then the second, *timed verification*, phase refines the untimed assumption into timed one and concludes the verification result. The flow is complete, i.e., users are guaranteed to get the verification result. Interested readers are referred to the technical report [7]. After verification, PAT shows that the I/O system satisfies the property φ.

3 Experimental Results and Discussion

To show the feasibility and scalability of our tool, we present verification results of four different applications, namely the CSS, GSS, FMS, and AIP systems, in Table 1. The details of the four systems, their models, and the verified properties can be found in [6]. The experimental results were obtained by running PAT on a Windows 7 machine with a 2.27 GHz Intel(R) Core(TM) i3 processor and 4 GB RAM. We also compared our approach with the UPPAAL model checker [11]; however, we do not list the verification time of UPPAAL for verifying the AIP system because UPPAAL does not support events on transitions such that the AIP system cannot be modeled in UPPAAL. When the system size is small, compositional approach does not outperform monolithic verification or UPPAAL because of the overhead of learning iterations; when the number of components increases, the learning iterations compensate for the large global state

Table 1. Verification Results

				Monolithic				Compositional				UPPAAL
System	n	$\lvert C_\Sigma\rvert$	$\frac{\lvert P_{\not\models}\rvert}{\lvert P\rvert}$	$\lvert L\rvert_{max}$	$\lvert\delta\rvert_{max}$	Time (secs)	Mem (MB)	$\lvert L\rvert_{max}$	$\lvert\delta\rvert_{max}$	Time (secs)	Mem (MB)	Time (secs)
CSS	3	6	0/6	11	20	0.03	0.16	19	50	0.06	0.77	0.05
GSS	3	3	2/3	29	46	0.03	0.13	56	107	0.03	0.69	0.06
FMS-1	5	3	1/3	193	514	0.03	1.18	60	138	0.03	0.89	0.08
FMS-2	10	6	3/6	76,305	396,789	40.71	114.08	1,492	4,952	0.66	6.60	2.05
FMS-3	11	6	5/7	201,601	1,300,566	70.02	295.89	3,150	16,135	1.14	12.07	9.87
FMS-4	14	8	3/9	–	–	–	ROM	26,320	127,656	51.02	41.41	ROM
AIP	10	4	5/10	104,651	704,110	78.05	149.68	2,992	12,971	1.90	7.39	N/A

n: # of components; $\lvert C_\Sigma\rvert$: # of event-recording clocks; $\lvert P\rvert$: # of properties; $\lvert P_{\not\models}\rvert$: # of violated properties; $\lvert L\rvert_{max}$: # of visited locations during verification; $\lvert\delta\rvert_{max}$: # of visited transitions during verification; ROM: run out of memory

space and compositional approach can reduce the verification time and the memory usage significantly. For the FMS-4 system, the monolithic approach and UPPAAL cannot even finish the verification using 4 GB memory.

Discussion. AGR has been applied to model checking to alleviate the state space explosion problem [3]. However, the construction of the assumptions for AGR usually requires nontrivial creativity and experience, which limits the impact of AGR. Cobleigh et al. [4] proposed a framework that generates the assumptions of components automatically using the L^* algorithm [2]. This work was a breakthrough of automating compositional verification for untimed systems. Grinchtein et al. [5] proposed three algorithms for learning ERAs; however, the time complexity of the algorithms depend exponentially on the largest constant appearing in the time constraints. In [8], we proposed a more efficient polynomial time algorithm, TL^*, for learning ERAs. Starting from 2010, ERA module in PAT has come to a stable stage with solid testing. We successfully applied it to verify real-time systems ranging from classical concurrent algorithms to real world problems. In the future, we plan to use different techniques to generate the assumptions and to extend the framework using other proof rules of AGR.

References

1. Alur, R., Fix, L., Henzinger, T.A.: Event-clock automata: A determinizable class of timed automata. Theoretical Computer Science 211(1-2), 253–273 (1999)
2. Angluin, D.: Learning regular sets from queries and counterexamples. Information and Computation 75(2), 87–106 (1987)
3. Clarke, E.M., Long, D.E., MacMillan, K.L.: Compositional model checking. In: LICS, pp. 353–362 (1989)
4. Cobleigh, J.M., Giannakopoulou, D., Păsăreanu, C.S.: Learning Assumptions for Compositional Verification. In: Garavel, H., Hatcliff, J. (eds.) TACAS 2003. LNCS, vol. 2619, pp. 331–346. Springer, Heidelberg (2003)
5. Grinchtein, O., Jonsson, B., Leucker, M.: Learning of event-recording automata. Theorectical Computer Science 411(47), 4029–4054 (2010)
6. Lin, S.W.: https://sites.google.com/site/shangweilin/era-pat
7. Lin, S.W.: https://sites.google.com/site/shangweilin/technical-reports

8. Lin, S.-W., André, É., Dong, J.S., Sun, J., Liu, Y.: An Efficient Algorithm for Learning Event-Recording Automata. In: Bultan, T., Hsiung, P.-A. (eds.) ATVA 2011. LNCS, vol. 6996, pp. 463–472. Springer, Heidelberg (2011)
9. Namjoshi, K.S., Trefler, R.J.: On the Completeness of Compositional Reasoning. In: Emerson, E.A., Sistla, A.P. (eds.) CAV 2000. LNCS, vol. 1855, pp. 139–153. Springer, Heidelberg (2000)
10. Sun, J., Liu, Y., Dong, J.S., Pang, J.: PAT: Towards Flexible Verification under Fairness. In: Bouajjani, A., Maler, O. (eds.) CAV 2009. LNCS, vol. 5643, pp. 709–714. Springer, Heidelberg (2009)
11. UPPAAL, http://www.uppaal.org/