Science of Computer Programming 113 (2015) 85–118

Contents lists available at ScienceDirect

Science of Computer Programming

www.elsevier.com/locate/scico

Formal semantics and efficient analysis of Timed Rebeca in Real-Time Maude

Zeynab Sabahi-Kaviani [a], Ramtin Khosravi [a], Peter Csaba Ölveczky [b,*], Ehsan Khamespanah [a], Marjan Sirjani [c]

[a] *School of Electrical and Computer Engineering, University of Tehran, Iran*
[b] *Department of Informatics, University of Oslo, Norway*
[c] *School of Computer Science, Reykjavik University, Iceland*

ARTICLE INFO

Article history:
Received 28 June 2014
Received in revised form 22 April 2015
Accepted 13 July 2015
Available online 29 July 2015

Keywords:
Real-time actors
Timed Rebeca
Formal semantics
Model checking
Real-Time Maude

ABSTRACT

The actor model is a well-established and intuitive model of distributed computation. Timed Rebeca is a timed extension of the actor-based modeling language Rebeca. Although Rebeca is supported by a rich verification toolset, Timed Rebeca has not had an executable formal semantics, and has therefore had limited support for formal analysis. In this paper, we provide a formal semantics for Timed Rebeca in Real-Time Maude. Our semantics exploits the isolation inherent in the actor model: since no actor can access the state variables of other actors, and since actors execute only one message server at a time, the effect of executing a statement is not observable by other actors. We can therefore "group together" a number of statements and execute them in one "atomic" rewrite step; this significantly improves the performance of model checking compared to standard approaches where each action is performed by a rewrite step. We have automated the translation from Timed Rebeca to Real-Time Maude, allowing Timed Rebeca models to be automatically analyzed using Real-Time Maude's reachability analysis tool and LTL and timed CTL model checkers. This enables a formal model-engineering methodology which combines the convenience of intuitive modeling in Timed Rebeca with formal verification in Real-Time Maude. We illustrate this methodology—and the performance gained by our new "partial-order-reduction-like" optimized semantics—with a number of case studies, including the IEEE 802.11 RTS/CTS collision avoidance protocol for wireless networks.

© 2015 Elsevier B.V. All rights reserved.

1. Introduction

The importance of formal modeling and analysis for ensuring the dependability and correctness of safety-critical systems has long been acknowledged. However, the lack of formal modeling languages close to programming and modeling languages used by practitioners has limited the use of formal methods. The actor model [1] is a well-established and intuitive model of distributed computation. The units of concurrency are called actors, each of which has its own execution thread. An actor encapsulates its state and communicates with other actors through message passing. Timed Rebeca [2] is

* Corresponding author.
E-mail addresses: z.sabahi@ece.ut.ac.ir (Z. Sabahi-Kaviani), r.khosravi@ut.ac.ir (R. Khosravi), peterol@ifi.uio.no (P.C. Ölveczky), e.khamespanah@ece.ut.ac.ir (E. Khamespanah), marjan@ru.is (M. Sirjani).

http://dx.doi.org/10.1016/j.scico.2015.07.003
0167-6423/© 2015 Elsevier B.V. All rights reserved.

an actor-based modeling language that extends the Rebeca language [3] to support the modeling of distributed real-time systems. Because of its Java-like syntax and its simple and intuitive message-driven and object-based computational model, Timed Rebeca should be an easy-to-learn language for system developers, thereby bridging the gap between formal methods and practical software engineering. It has been applied to several application domains such as modeling and analysis of routing algorithms for Network-on-Chips (NOCs) [4,5], and schedulability analysis of wireless sensor and actuator network applications (specifically, real-time continuous sensing application for structural health monitoring).

Although Rebeca is supported by a rich model checking toolset [6], Timed Rebeca has had limited support for formal analysis. Even though Timed Rebeca has an SOS semantics, it lacks an executable formal semantics that would enable automated analysis methods such as simulation and temporal logic model checking.

However, providing an executable formal semantics for Timed Rebeca is quite challenging. For example, since Timed Rebeca has a rich expression/statement language that allows the values of state variables to grow beyond any bound, since the message queues can become arbitrarily long, and since Timed Rebeca supports the dynamic creation of new actors, Timed Rebeca cannot be translated into popular real-time formalisms such as, e.g., timed automata [7].

In this paper, we provide a formal Real-Time Maude semantics for Timed Rebeca. Real-Time Maude [8,9] is a specification formalism and analysis tool for real-time systems based on rewriting logic [10]. With its natural time model and expressive formalism, which is particularly suitable for formally specifying distributed real-time systems in an object-oriented way, Real-Time Maude should be ideally suited for this challenging task. Real-Time Maude is supported by a high-performance toolset providing a spectrum of analysis methods, including simulation through timed rewriting, reachability analysis, and (untimed) linear temporal logic (LTL) model checking as well as timed CTL model checking.

We have automated the translation from Timed Rebeca to Real-Time Maude, so that the user gets Real-Time Maude simulation and model checking of her Timed Rebeca model for free. Furthermore, such formal analysis has been integrated into the Eclipse-based Afra modeling environment and toolset for Rebeca. This would of course not be very useful if the user would need to understand the Real-Time Maude representation of her Timed Rebeca model, and/or would need to define state properties in Real-Time Maude, in order to model check her Timed Rebeca model. We have therefore taken advantage of Real-Time Maude's support for *parametric* state propositions to predefine useful generic state propositions, so that the user can define her (possibly timed) temporal logic properties *without* having to know Real-Time Maude or understand how the mapping from Timed Rebeca works. Altogether, this enables a formal model-engineering methodology that combines the convenience of modeling in an intuitive actor language with Java-like syntax with formal verification in Real-Time Maude.

This paper is an extended version of the paper [11]. Apart from adding much more detail on the formal semantics of Timed Rebeca, this paper extends [11] as follows:

1. The dynamic creation of new actors is now supported. Furthermore, by passing around the actor identifiers, we now fully support systems with dynamic topology.
2. We have developed a new "partial-order-reduction-based" optimized Real-Time Maude semantics, where each actor executes *all its consecutive "immediate" actions* in *one step*, instead of using one step for each action, thereby avoiding many unnecessary interleavings.
3. We compare the performance of model checking using the "standard" semantics and our new semantics. This comparison shows a drastic performance improvement which makes model checking analysis feasible even when such analysis is unfeasible with our previous semantics.
4. We now discuss the soundness and completeness of the Real-Time Maude analyses of Timed Rebeca models. The point is that, although Timed Rebeca assumes discrete time, it is for *efficiency purposes* much better to "fast-forward" time advance to the next moment when an "event" takes place than to visit *each* moment in time during Real-Time Maude analysis. However, Real-Time Maude analysis using such "time-fast-forwarding" is *in general* not sound and complete since not all behaviors (e.g., those obtained by stopping time also when no event takes place) are analyzed. Nevertheless, as discussed in Section 5.2, soundness/completeness of this efficient model checking strategy is indeed guaranteed for most formulas occurring in practice. (The source of the incompleteness is our efficient modeling of the removal of messages whose deadlines have expired.) Furthermore, soundness and completeness is guaranteed if the Real-Time Maude model checking of a Timed Rebeca model uses the less efficient strategy that visits all instants in time.
5. We add three new applications: a small ticket service system, a sensor system to measure toxic gas in an environment, and a dynamic client/server architecture which requires dynamic actor creation and dynamic topologies.
6. We explain the model development and verification toolchain TR2RTM and how to use it.

The rest of the paper is structured as follows. Section 2 provides necessary background on Real-Time Maude. Section 3 gives an overview of Timed Rebeca. Section 4 explains the Real-Time Maude formalization of the Timed Rebeca semantics. Section 5 defines some useful generic atomic state propositions that allows the user to easily define her temporal logic formulas without knowing Real-Time Maude, and discusses soundness and completeness of Real-Time Maude model checking of Timed Rebeca models. Section 5 also explains how Real-Time Maude analysis has been integrated into the Rebeca toolset. Section 6 illustrates our methodology on several applications. Section 7 compares the performance of model checking the Real-Time Maude models obtained by using, respectively, our previous semantics and our new semantics. Finally, Section 8 discusses related work and Section 9 gives some concluding remarks.

Download English Version:

https://daneshyari.com/en/article/434009

Download Persian Version:

https://daneshyari.com/article/434009

Daneshyari.com