Quantitative Evaluation in Embedded System Design: Trends in Modeling and Analysis Techniques

Joost-Pieter Katoen
RWTH Aachen University, Aachen, Germany
katoen@cs.rwth-aachen.de

Abstract

The evaluation of extra-functional properties of embedded systems, such as reliability, timeliness, and energy consumption, as well as dealing with uncertainty, e.g., in the timing of events, is getting more and more important. What are the models and approaches to analyze such properties in a reliable way? We survey some main developments and trends in the modeling, and the analysis of these aspects and stress the importance of approaches that tackle both extra-functional, as well as correctness aspects.

1 Introduction

Embedded systems are subject to complex and permanent interactions with their – mostly physical – environment via sensors and actuators. Typically, embedded systems do not terminate and interaction usually takes place with multiple concurrent processes at the same time. Reactions to the stimuli provided by the environment should be prompt (responsiveness), i.e., the system has to "keep up" with the speed of the processes with which it interacts. Due to its safety-critical character, the integrity (i.e., correctness) of generated outputs is vital. As embedded software executes on devices where several other activities go on, non-functional properties such as efficient usage of resources (e.g., power consumption) and robustness are important. High requirements are put on performance and dependability, since the embedded nature complicates tuning and maintenance.

Errors in embedded systems can be costly and catastrophic. Dramatic examples from the recent past are known. The fatal defects in the control software of the Ariane-5 missile and the Mars Pathfinder are renowned by now. The bug in Intel's Pentium-II floating-point division unit in the early 1990s caused a huge financial loss, and severely damaged Intel's reputation. Correctness of embedded software is thus of vital importance.

Our Focus In order to capture errors and quantitative aspects as early as possible in the design trajectory we focus on *model-driven system development*. Ideally, this allows engineers to (graphically) model the requirements, behaviour and functionality of embedded systems. We firmly believe that a *highly integrated approach* toward the modeling and analysis of functional, as well as extra-functional aspects is essential within this model-driven design framework. A single model that captures both aspects in a coherent manner avoids the presence of inconsistensies that can easily arise when multiple systems views are spread over various different models. In addition, correctness and extra-functional aspects are inextrably related. As most (if not all) quantitative aspects of embedded systems are subject to random phenomena, e.g., failures of components, response delays of environment, and battery depletion, our focus is on techniques in which stochastic elements are dominant as opposed to e.g., models that use nondeterministic timing.

2 Modeling Techniques

Standard Models A large variety of stochastic models exists that mainly differ in the assumed distributions that determine state residence times. Markov chains are one of the most popular models for the evaluation of performance and dependability of information processing systems. To obtain performance measures, typically long-run or transient state probabilities of Markov chains are determined. Sometimes the Markov chain at hand is equipped with rewards and computations involve determining long-run or instantaneous reward probabilities. State residence times are mostly assumed to be exponential, although generalizations exist that are more liberal. Exponential distributions are amenable to efficient numerical analysis and can be used for many real-life phenomena. More general models are typically analyzed by simulation. To avoid the specification of Markov chains directly at the state level, high-level model specification techniques have been developed, most notably those based on queueing networks, Petri nets, and stochastic activity networks. With appropriate software tools support-

978-3-9810801-3-1/DATE08 © 2008 EDAA

ing these specification methods, it is relatively comfortable to specify performance and dependability models of which the underlying models have millions of states.

Need for Nondeterminism Timeliness and interaction with the environment are main characteristics of embedded software. Stochastic models that faithfully represent embedded software thus need to be timed and open. Openness entails that the behavior of the environment is underspecified, and that interactions may be abstracted from. Nondeterminism is the technique par excellence for this purpose. In addition, nondeterminism is essential for modeling concurrency and as a means for abstraction where detailed models are "collapsed" into smaller models. Extensions of discrete and timed Markov chains that exhibit nondeterminism, and that are amenable to compositional modeling are interactive Markov chains [4] and probabilistic automata [5]. These models have strong similarities with Markov decision processes, models that are of major importance in stochastic operations research, and automated planning (e.g., robot control) in artificial intelligence.

Compositional Modeling Modeling large stochastic discrete-event dynamic systems is a difficult task that typically requires human intelligence and ingenuity. To facilitate this modeling process, formalisms have been developed that allow for modeling such systems in a *compositional* manner. This allows to construct models of simpler components—usually from first principles—that can be combined by appropriate composition operators to yield complete system models. This property enables to enrich existing untimed specifications with random timing constraints by just composition. The description of time constraints can thus takes place in a *modular* way, that is, as separated processes that are constraining the behavior by running in parallel with an untimed (or otherwise time-constrained) process [4]. Such approaches provide an apparatus for compositional reasoning about the structure and behavior of systems, and features abstraction mechanisms enabling the treatment of system components as black boxes. More importantly, though, these formalisms can be equipped with behavioural equivalences that, under mild conditions, are substitutive, i.e., models can be simplified in a component-wise manner. Together with efficient algorithms this allows for the component-wise minimization of system models This paradigm originates from the field of process algebras such as CCS, CSP, and LOTOS and has been successfuly extended with stochastic aspects in the last decade [3]. Lately, attempts have been persued to link these combined approaches to system design languages, such as AADL, fault trees [2], UML Statecharts, and VHDL. Interestingly enough, nondeterminism is a key ingredient to enable the compositional modeling.

3 Analysis Techniques

Classical analysis techniques for extra-functional properties range from numerical, analytical, to simulative approaches. Popular industrial techniques for checking the correctness of designs are peer review and simulation. In practice, more time and effort are spent on verification than on construction. Techniques are sought to reduce and ease the verification efforts while increasing their coverage. Formal methods such as model checking offer a large potential to obtain an early integration of verification in the design process, to provide more effective verification techniques, and to reduce the verification time. More importantly, though, recent advancements to model checking provide ample means to determine performance and dependability *guarantees* of stochastic models.

Properties are specified in a notation that has its roots in logics, and allows for the specification of functional properties such as safety ("is the system never in a bad state"), liveness ("is the system eventually in a good state"), and fairness constraints that typically rule out unrealistic system behaviours, while, in addition, typical extra-functional properties can be specified in an unambiguous and lucid manner. The use of logics yields an expressive framework that allows to express well-known measures, but also (new) intricate and complex performance guarantees. Given a precise description of the desired guarantee, all states in the Markov chain are determined that surely meet the property. This is done in a fully automated way. The power of this technique is that no matter how complex the logical guarantee, it is *automatically* checked which states in the Markov chain satisfy it. Neither manual manipulations of models (or their high-level descriptions) are needed, nor the knowledge of any numerical technique to analyze them efficiently. This applies to any Markov chain of any structure specified in any high-level formalism [1]. These advantages have led to the adoption of model checking for quantitative analysis in several tools such as CADP, Statemate, and GreatSPN.

References

[1] C. Baier, B. R. Haverkort, H. Hermanns, and J.-P. Katoen. Model checking meets performance evaluation. *ACM Perf. Ev. Review*, 32(4):10–15, 2005.

[2] H. Boudali, P. Crouzen, and M. Stoelinga. A compositional semantics for dynamic fault trees in terms of interactive Markov chains. In *ATVA*. Springer Verlag, 2007.

[3] M. Bravetti, H. Hermanns, and J.-P. Katoen. Ymca: - why Markov chain algebra? *ENTCS*, 2(162):107–112, 2006.

[4] H. Hermanns and J.-P. Katoen. Automated compositional Markov chain generation for a plain-old telephone system. *Sci. of Comp. Progr.*, 36(1):97–127, 2000.

[5] R. Segala and N. A. Lynch. Probabilistic simulations for probabilistic processes. *Nord. J̇Comput.*, 2(2):250–273, 1995.