Analysis and Tools: Abstraction and Compositionality

KUN

May 1, 2004

AMETIST DELIVERABLE 2.1.1

Project acronym: AMETIST
Project full title: Advanced Methods for Timed Systems
Project no.: IST-2001-35304
Project Co-ordinator: Frits Vaandrager
Project Start Date: 1 April 02
Duration: 36 months
Project home page: http://ametist.cs.utwente.nl/

In Task 2.1, the Ametist consortium studies the application of abstraction, compositionality and structure exploitation as key techniques in controlling and reducing the complexity involved in analyzing real-time system models.

Abstraction and compositionality are well established for model checking and theorem proving approaches to discrete state system verification. *Abstraction* — either based on generic principles (like from timed to untimed systems, infinite data domains to finite one) or on case-specific user suggested simplifications — serves to simplify the models by omitting aspects that are not relevant for analysis. *Compositionality* provides a divide-and-conquer approach to manage complexity: properties of complicated systems are inferred from properties of their components. For the verification of finite-state systems a number of techniques for *exploiting the composite structure* of the model exists. Partial order reduction and compositional backwards reachability are methods which help to reduce search spaces for loosely coupled concurrent components. Likewise, methods exploiting symmetries and hierarchical structure have shown to be very successful.

This deliverable presents a brief overview of the results obtained by Ametist that relate to abstraction and compositionality. A discussion of the (impressive) results on structure exploitation will be deferred to Deliverable 2.1.2, that is due next year. In fact, most of the efforts in Task 2.1 thus far have been spent on developing techniques for structure exploitation (18 papers), a significant effort has been devoted to abstraction (11 papers), whereas compositionality hardly received any attention (1 paper). The improvement in terms of performance of timed automata tools that has resulted from the progress in the areas of abstraction and structure exploitation (in particular partial order methods and symmetry reduction) is far beyond expection (see Deliverable 2.5.a for some statistics). Consequently, these areas have drawn most attention from Ametist researchers during the initial 2 years of the project.

Abstraction Based Techniques

Timed Automata

Ametist researchers have made significant progress by carefully reexamining and improving the traditional zone abstraction for timed automata.

Since about 10 years, several tools implement the timed automata model and are successfully used to verify real-life examples. In spite of this well-established framework, [4] proves that the forward analysis algorithm implemented in these tools is not correct! However, the paper also proves that it is correct for a restricted class of timed automata, which has been sufficient for modeling numerous real-life systems.

By definition timed automata have an infinite state-space, thus for verification purposes, an exact finite abstraction is required. In [2], we propose a location based finite zone abstraction, which computes an abstraction based on the relevant guards for a particular state of the model (as opposed to all guards). We show that the location-based zone abstraction is sound and complete with respect to location reachability; that it generalises active-clock reduction, in the sense that an inactive clock has no relevant guards at all; that it enlarges the class of timed automata, that can be verified. We generalise the new abstraction to the case of networks of timed automata, and experimentally demonstrate a potentially exponential speedup compared to the usual abstraction.

Timed automata have an infinite semantics. For verification purposes, one usually uses zone based abstractions w.r.t. the maximal constants to which clocks of the timed automaton are compared. Paper [3] shows that by distinguishing maximal lower and upper bounds, significantly coarser abstractions can be obtained. The paper shows soundness and completeness of the new abstractions w.r.t. reachability. It demonstrates how information about lower and upper bounds can be used to optimise the algorithm for bringing a difference bound matrix into normal form. Finally, the paper experimentally demonstrate that the new techniques dramatically increases the scalability of the real-time model checker Uppaal.

Hybrid Automata

Without aggressive abstractions it is virtually impossible to verify interesing properties of hybrid systems. A very interesting recent line of research aims at successive refinement of abstractions via counterexamples in model checking of hybrid systems.

Papers [7, 10, 6, 9] study abstraction and counterexample-guided refinement in model checking of hybrid systems.

Hybrid dynamic systems include both continuous and discrete state variables. Properties of hybrid systems, which have an infinite state space, can often be verified using ordinary model checking together with a finite-state abstraction. Model checking can be inconclusive, however, in which case the abstraction must be refined. In [7, 10, 6, 9], we present a new procedure to perform this refinement operation for abstractions of hybrid systems. Following an approach originally developed for finite-state systems, the refinement procedure constructs a new abstraction that eliminates a counterexample generated by the model checker. For hybrid systems, analysis of the counterexample requires the computation of sets of reachable states in the continuous state space. We show how such reachability computations with varying degrees of complexity can be used to refine hybrid system abstractions efficiently. Examples illustrate our counterexample-guided refinement procedure. Experimental results for a prototype implementation indicate significant advantages over existing methods.

In [11], [[I need the abstract!]

Computing reachable sets is an essential step in most analysis and synthesis techniques for hybrid systems. The representation of these sets has a deciding impact on the computational complexity and thus the applicability of these techniques. In [11], we present a new approach for approximating reachable sets using oriented rectangular hulls (ORHs), the orientations of which are determined by singular value decompositions of sample covariance matrices for sets of reachable states. The orientations keep the over-approximation of the reachable sets small in most cases with a complexity of low polynomial order with respect to the dimension of the continuous state space. We show how the use of ORHs can improve the efficiency of reachable set computation significantly for hybrid systems with nonlinear continuous dynamics.

In [1], we present an abstraction method for nonlinear continuous systems. The main idea of our method is to project out some continuous variables, say z, and treat them in the dynamics of the remaining variables x as uncertain input. Therefore, the dynamics of x is then described by a differential inclusion. In addition, in order to avoid excessively conservative abstractions, the domains of the projected variables are divided into smaller regions corresponding to different differential inclusions. The final result of our abstraction procedure is a hybrid system of lower dimension with some important properties that guarantee convergence results. The applicability of this abstraction approach depends on the ability to deal with differential inclusions. We then focus on uncertain bilinear systems, a simple yet useful class of nonlinear differential inclusions, and develop a reachability technique using optimal control. The combination of the abstraction method and the reachability analysis technique for bilinear systems allows to treat multi-affine systems, which is illustrated with a biological system.

Methodology of Model Checking

In [8], we take a closer look at the automated analysis of designs, in particular of verification by model checking. Model checking tools are increasingly being used for the verification of real-life systems in an industrial context. In addition to ongoing research aimed at curbing the complexity of dealing with the inherent state space explosion problem - which allows us to apply these techniques to ever larger systems - attention must now also be paid to the methodology of model checking, to decide how to use these techniques to their best advantage. Model checking in the large causes a substantial proliferation of interrelated models and model checking sessions that must be carefully managed in order to control the overall verification process. We show that in order to do this well both notational and tool support are required. We discuss the use of software configuration management techniques and tools to manage and control the verification trajectory. We present Xspin/Project, an extension to Xspin, which automatically controls and manages the validation trajectory when using the model checker Spin.

Computer-aided verification of embedded systems hinges on the availability of good verification models of the systems at hand. Because of the combinatorial complexities that are inherent in any process of verification, such models generally are only abstractions of the full design model or system specification. As they must both be small enough to be effectively verifiable and preserve the properties under verification, the development of verification models usually requires the experience, intuition and creativity of an expert. We argue that there is a great need for systematic methods for the construction of verification models to move on, and leave the current stage that can be characterised as that of "model hacking". The ad-hoc construction of verification models obscures the relationship between models and the systems that they represent, and undermines the reliability and relevance of the verification results that are obtained. In [5], we propose some ingredients for a solution to this problem.

Compositionality Based Techniques

In [12], a scheduler synthesis is described for the memory interface card offered by Terma as a case study for the AMETIST consortium. A model of the system in the SMV modeling language is introduced. This model can be used to produce valid schedules for systems with small number of buffers. A method to synthesize an optimal (in terms of buffer sizes) scheduler for the full system is also given. The schedule is verified by the SMV model checker using the above-mentioned model. Some extensions to cards with other parameters are discussed.

The main analysis described in [12] is highly compositional. Instead of analyzing the whole system as a network of interacting automata, the author chooses to analyze each component at a time. The result of this analysis is a characterization of all stable loops of the component (in the specific case - the possible sequences of times between refreshes). With a characterization of the possible patterns of usage of the bus for all the elements, the author was able to design an optimal schedule. The paper can be considered as an example of how compositional reasoning can be exploited in the area of scheduling, and may evolve to a more general technique.

A substantial contribution to compositional verification is scheduled to appear this year when Goran Frehse expects to complete and defend his PhD thesis (prepared in Uni DO and to be defended in KUN) on compositional verification of hybrid systems using simulation relations. An interesting question is to which extend the assume/guarantee techniques from this thesis can also be effectively applied in a setting of timed automata.

References

[1] Eugene Asarin and Thao Dang. Abstraction by projection and application to multi-affine systems. In Rajeev Alur and George Pappas, editors, *Hybrid Systems: Computation and Control Proceedings of 7th International Workshop*, volume 2993 of *LCNS*, Philadelphia, PA, USA, 2004.

[2] G. Behrmann, P. Bouyer, E. Fleury, and K. G. Larsen. Static guard analysis in timed automata verification. In *Proc. 9th Int. Conf. Tools and Algorithms for the Construction and Analysis of Systems (TACAS'2003)*, volume 2619 of *Lecture Notes in Computer Science*, pages 254–277. Springer-Verlag, 2003. Available from World Wide Web: `http://www.lsv.ens-cachan.fr/Publis/PAPERS/BBFL-tacas-2003.ps`.

[3] G. Behrmann, P. Bouyer, K. G. Larsen, and R. Pelánek. Lower and upper bounds in zone based abstractions of timed automata. In *Proc. 10th Int. Conf. Tools and Algorithms for the Construction and Analysis of Systems (TACAS'2004)*, volume 2988 of *Lecture Notes in*

Computer Science, pages 312–326. Springer-Verlag, 2004. Available from World Wide Web: `http://www.lsv.ens-cachan.fr/Publis/PAPERS/BBLP-tacas04.ps`.

[4] P. Bouyer. Untamable timed automata. In *Proc. of 20th Ann. Symp. on Theoretical Aspects of Computer Science (STACS'2003)*, volume 2607 of *Lecture Notes in Computer Science*, pages 620–631. Springer-Verlag, 2003. Available from World Wide Web: `http://www.lsv.ens-cachan.fr/Publis/PAPERS/Bou-stacs2003.ps`.

[5] Ed Brinksma and Angelika Mader. On verification modelling of embedded systems. In *SEES 2003: Software Engineering for Embedded Systems: from Requirements to Implementation*, Monterey Workshop Proceedings, pages 101–105, 2004.

[6] E. Clarke, A. Fehnker, Z. Han, B. H. Krogh, O. Stursberg, and M. Theobald. Verification of hybrid systems based on counterexample-guided abstraction refinement. In *Tools and Algorithms for the Construction and Analysis of Systems, LNCS 2619*, pages 192–207. Springer, 2003. Available from World Wide Web: `http://ametist.cs.utwente.nl/INTERNAL/PUBLICATIONS/DORTMUNDPublications/TACAS_03.pdf`.

[7] E. Clarke, A. Fehnker, Z. Han, B.H. Krogh, J. Ouaknine, O. Stursberg, and M. Theobald. Abstraction and counterexample-guided refinement in model checking of hybrid systems. *Int. Journal Foundations of Computer Science*, 14(4):583–604, 2003. Available from World Wide Web: `http://ametist.cs.utwente.nl/INTERNAL/PUBLICATIONS/DORTMUNDPublications/IJFCS03.pdf`.

[8] Theo Ruys and Ed Brinksma. Managing the verification trajectory. *STTT*, 4(2):246–259, 2003. Available from World Wide Web: `http://springerlink.metapress.com/openurl.asp?genre=article&id=doi:10.1007/s10009-002-0078-1`.

[9] O. Stursberg, A. Fehnker, Z. Han, and B. H. Krogh. Specification-guided analysis of hybrid systems using a hierarchy of validation methods. In *IFAC Conf. on Analysis and Design of Hybrid Systems*, pages 289–295, 2003. Available from World Wide Web: `http://ametist.cs.utwente.nl/INTERNAL/PUBLICATIONS/DORTMUNDPublications/adhs_cgv.pdf`.

[10] O. Stursberg, A. Fehnker, Z. Han, and B.H. Krogh. Verification of a cruise control system using counterexample-guided search. *Control Engineering Practice*, 2004. Available from World Wide Web: `http://ametist.cs.utwente.nl/INTERNAL/PUBLICATIONS/DORTMUNDPublications/CEP04a.pdf`. (to appear).

[11] O. Stursberg and B. H. Krogh. Efficient representation and computation of reachable sets for hybrid systems. In *Hybrid Systems: Computation and Control, LNCS 2623*, pages 482–497. Springer, 2003. Available from World Wide Web: `http://ametist.cs.utwente.nl/INTERNAL/PUBLICATIONS/DORTMUNDPublications/hscc03_hull.pdf`.

[12] Gera Weiss. Optimal Scheduler for a Memory Card. Research report, Weizmann, 2002. Available from World Wide Web: `http://ametist.cs.utwente.nl/INTERNAL/PUBLICATIONS/WISPublications/Optimal_Schedule_for_a_Memory_Card`.