How to Prove Hybrid Systems and Why that Matters

Abstract for Invited Talk

André Platzer
Logical Systems Lab
Computer Science Department
Carnegie Mellon University
Pittsburgh, USA
aplatzer@cs.cmu.edu

***Abstract*—This invited talk provides a brief exposition how hybrid systems proving works, why hybrid systems verification is an important device to ensure the safety of complex systems, and gives an idea where that technology is successful.**

I. Overview

Hybrid systems are the most important mathematical model for embedded systems and are just as relevant for cyber-physical systems. Developing correct controllers for these systems is quite challenging but worth the time investment, because of the benefits that computer control principles can have on the applications. Verification of hybrid systems provides a way of obtaining those benefits with guarantees about the safety and behavior of the system [1]. The verification tools KeYmaera [2] and its successor KeYmaera X [3] implement differential dynamic logic [4], [5], a logic that is directly suitable for systematic verification and reasoning for hybrid systems. These tools have been used for verifying several complex applications, including the Airborne Collision Avoidance System ACAS X, the European Train Control System ETCS, several automotive systems, mobile robot navigation with the dynamic window algorithm, and a surgical robotic system for skull-base surgery. This invited talk gives a brief exposition of the approach and shows where and why and how it has been used successfully in applications.

For more surveys and more details, we refer to the literature [1], [5]–[9].

II. Hybrid Systems

Hybrid systems [10], [11] are mathematical models that feature mixed discrete as well as continuous dynamics. Among other applications, hybrid systems are crucial for understanding the behavior of embedded systems as well as cyber-physical systems. Canonically, discrete dynamics comes from decisions and computer program execution while continuous dynamics comes from physical motion, e.g., of a car or an aircraft. Hybrid programs are a programming language for hybrid systems [4], [5]. They can be thought of as adding differential equations as a new form of statements to conventional discrete programming languages.

Conventional programming languages are the natural model for describing discrete computer programs but are not particularly capable when it comes to modeling continuous behavior.[1] Control theory has found differential equations to be a powerful language for continuous dynamics, but they are not a good match for understanding the sudden changes of discrete computation.

Hybrid programs provide both sources of dynamics, discrete programs as well as differential equations, within a single programming language. That makes it possible to leverage the structuring mechanisms and compositionality features of programming languages without losing the descriptive power of differential equations for continuous behaviors.

III. Formal Specifications

Describing the system models of interest as a hybrid program is the first step. Precisely specifying the desired correctness properties is the second step. KeYmaera and KeYmaera X provide differential dynamic logic [4], [5], [13] as a language for formally specifying and verifying correctness properties of hybrid systems. For example, formulas of the form

$$\phi \to [\alpha]\psi$$

express that all states reachable by the behavior of the hybrid program α from initial states satisfying the assumption ϕ satisfy the postcondition ψ. This is a logical formula expressing a Hoare-triple $\{\phi\}\alpha\{\psi\}$ but for hybrid system α instead of merely for a conventional discrete program α. Arbitrary other combinations of these and other logical connectives are supported as well that are not expressible as Hoare-triples.

IV. Applications

KeYmaera and KeYmaera X have been used for verifying a number of interesting applications, including Airborne Collision Avoidance System ACAS X [14], [15], roundabout type aircraft maneuvers [16], the European Train Control System ETCS [17], several automotive systems (e.g., provably safe adaptive cruise controllers for cars on highways [18]), mobile robot navigation with the dynamic window algorithm [19], and a surgical robotic system for skull-base surgery [20].

The Airborne Collision Avoidance System ACAS X, for example, has been subjected to a formal verification study [14], [15] using differential dynamic logic proofs. ACAS X

[1]A surprisingly close connection between discrete and continuous programs as well as hybrid programs exists regardless [8], [12].

© IEEE 2015
2015 International Conference on Complex Systems Engineering (ICCSE).
DOI: 10.1109/ComplexSys.2015.7385983

is a challenging industrial system, a canonical hybrid system in principle, but—with its half a trillion discrete states—overwhelmingly large. The verification, thus, consists of two phases, first identifying and verifying which collision avoidance advisory is safe under which circumstance in the hybrid systems model and then comparing the provably correct generic answer with the concrete ACAS X decisions. The overwhelming majority of the cases (97.7%) led to a provably safe collision advisory or unresolvable cases. The remaining 15,160,434,734 cases led to counterexamples, which are currently being investigated further because some of them can be resolved by follow-up advisories [15].

V. ModelPlex

An orthogonal but equally important aspect of KeYmaera or KeYmaera X is its support for ModelPlex [21]. ModelPlex is somewhat similar to Simplex [22] but for models instead of controllers. It does provide fundamentally stronger guarantees, though, in the form of fully rigorous proofs. ModelPlex is a systematic technique to generate provably correct monitor conditions that, if checked to hold at runtime, are provably guaranteed to imply that the offline safety verification results about a CPS model apply to the present run of the actual CPS implementation. Safety verification for CPS models, e.g., with hybrid systems models, is indubitably crucial to get complex systems with subtle interactions safe. Yet, the verification results about a CPS model only apply to the actual CPS to the extent that the model was accurate. While more modeling can help lead to higher fidelity results, this does not ultimately solve the Gordian Knot of models for models for CPSs. Unless one uses ModelPlex to cut through the Gordian Knot and finally translate safety guarantees about CPS models in a provably correct way to the safety of the particular run of a CPS implementation. The ModelPlex synthesis procedure itself is, indeed, also based on a nonclassical use of the proof procedures of differential dynamic logic.

Acknowledgment

This material is based upon work supported by the National Science Foundation under NSF CAREER Award CNS-1054246, NSF EXPEDITION CNS-0926181, and under Grant Nos. CNS-1035800 and CNS-0931985, by DARPA under agreement number FA8750-12-2-0291, by the ONR award N00014-10-1-0188, by the Army Research Office under Award No. W911NF-09-1-0273, and by the German Research Council (DFG) as part of the Transregional Collaborative Research Center "Automatic Verification and Analysis of Complex Systems" (SFB/TR 14 AVACS). These results are also supported in part by Carnegie Mellon University's Technologies for Safe and Efficient Transportation, The National USDOT University Transportation Center for Safety (T-SET UTC) which is sponsored by the US Department of Transportation.

References

[1] R. Alur, "Formal verification of hybrid systems," in *EMSOFT*, S. Chakraborty, A. Jerraya, S. K. Baruah, and S. Fischmeister, Eds. ACM, 2011, pp. 273–278.

[2] A. Platzer and J.-D. Quesel, "KeYmaera: A hybrid theorem prover for hybrid systems." in *IJCAR*, ser. LNCS, A. Armando, P. Baumgartner, and G. Dowek, Eds., vol. 5195. Springer, 2008, pp. 171–178.

[3] N. Fulton, S. Mitsch, J.-D. Quesel, M. Völp, and A. Platzer, "KeYmaera X: An axiomatic tactical theorem prover for hybrid systems," in *CADE*, ser. LNCS, A. Felty and A. Middeldorp, Eds., vol. 9195. Springer, 2015, pp. 527–538.

[4] A. Platzer, "Differential dynamic logic for hybrid systems." *J. Autom. Reas.*, vol. 41, no. 2, pp. 143–189, 2008.

[5] ——, "Logics of dynamical systems," in *LICS*. IEEE, 2012, pp. 13–24.

[6] ——, *Logical Analysis of Hybrid Systems: Proving Theorems for Complex Dynamics*. Heidelberg: Springer, 2010.

[7] ——, "Logic and compositional verification of hybrid systems (invited tutorial)," in *CAV*, ser. LNCS, G. Gopalakrishnan and S. Qadeer, Eds., vol. 6806. Springer, 2011, pp. 28–43.

[8] ——, "Analog and hybrid computation: Dynamical systems and programming languages," *Bulletin of the EATCS*, vol. 114, 2014.

[9] L. Doyen, G. Frehse, G. J. Pappas, and A. Platzer, "Verification of hybrid systems," in *Handbook of Model Checking*, E. M. Clarke, T. A. Henzinger, and H. Veith, Eds. Springer, 2016, ch. 28.

[10] A. Nerode and W. Kohn, "Models for hybrid systems: Automata, topologies, controllability, observability," in *Hybrid Systems*, ser. LNCS, R. L. Grossman, A. Nerode, A. P. Ravn, and H. Rischel, Eds., vol. 736. Springer, 1992, pp. 317–356.

[11] R. Alur, C. Courcoubetis, T. A. Henzinger, and P.-H. Ho, "Hybrid automata: An algorithmic approach to the specification and verification of hybrid systems." in *Hybrid Systems*, ser. LNCS, R. L. Grossman, A. Nerode, A. P. Ravn, and H. Rischel, Eds., vol. 736. Springer, 1992, pp. 209–229.

[12] A. Platzer, "The complete proof theory of hybrid systems," in *LICS*. IEEE, 2012, pp. 541–550.

[13] ——, "A uniform substitution calculus for differential dynamic logic," in *CADE*, ser. LNCS, A. Felty and A. Middeldorp, Eds., vol. 9195. Springer, 2015, pp. 467–481.

[14] J. Jeannin, K. Ghorbal, Y. Kouskoulas, R. Gardner, A. Schmidt, E. Zawadzki, and A. Platzer, "A formally verified hybrid system for the next-generation airborne collision avoidance system," in *TACAS*, ser. LNCS, C. Baier and C. Tinelli, Eds., vol. 9035. Springer, 2015, pp. 21–36.

[15] ——, "Formal verification of ACAS X, an industrial airborne collision avoidance system," in *EMSOFT*, A. Girault and N. Guan, Eds. IEEE Press, 2015, pp. 127–136.

[16] A. Platzer and E. M. Clarke, "Formal verification of curved flight collision avoidance maneuvers: A case study," in *FM*, ser. LNCS, A. Cavalcanti and D. Dams, Eds., vol. 5850. Springer, 2009, pp. 547–562.

[17] A. Platzer and J.-D. Quesel, "European Train Control System: A case study in formal verification," in *ICFEM*, ser. LNCS, K. Breitman and A. Cavalcanti, Eds., vol. 5885. Springer, 2009, pp. 246–265.

[18] S. M. Loos, A. Platzer, and L. Nistor, "Adaptive cruise control: Hybrid, distributed, and now formally verified," in *FM*, ser. LNCS, M. Butler and W. Schulte, Eds., vol. 6664. Springer, 2011, pp. 42–56.

[19] S. Mitsch, K. Ghorbal, and A. Platzer, "On provably safe obstacle avoidance for autonomous robotic ground vehicles," in *Robotics: Science and Systems*, P. Newman, D. Fox, and D. Hsu, Eds., 2013.

[20] Y. Kouskoulas, D. W. Renshaw, A. Platzer, and P. Kazanzides, "Certifying the safe design of a virtual fixture control algorithm for a surgical robot," in *HSCC*, C. Belta and F. Ivancic, Eds. ACM, 2013, pp. 263–272.

[21] S. Mitsch and A. Platzer, "ModelPlex: Verified runtime validation of verified cyber-physical system models," in *RV*, ser. LNCS, B. Bonakdarpour and S. A. Smolka, Eds., vol. 8734. Springer, 2014, pp. 199–214.

[22] D. Seto, B. Krogh, L. Sha, and A. Chutinan, "The Simplex architecture for safe online control system upgrades," in *ACC*, vol. 6, 1998, pp. 3504–3508.

[23] *Proceedings of the 27th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2012, Dubrovnik, Croatia, June 25–28, 2012*. IEEE, 2012.

[24] A. Felty and A. Middeldorp, Eds., *International Conference on Automated Deduction, CADE'15, Berlin, Germany, Proceedings*, ser. LNCS, vol. 9195. Springer, 2015.

[25] R. L. Grossman, A. Nerode, A. P. Ravn, and H. Rischel, Eds., *Hybrid Systems*, ser. LNCS, vol. 736. Springer, 1993.