Connecting Safety Verification with Implementable Hybrid Programs

Position Paper for the 2014 NSF Young Professionals Workshop on Exploring New Frontiers in Cyber-Physical Systems

Sarah M. Loos

Carnegie Mellon University
Computer Science Department
Pittsburgh, PA, USA
sloos@cs.cmu.edu

Keywords: hybrid programs, differential dynamic logic, formal verification, sequent calculus, theorem proving, CPS education

1 Introduction

Cyber-Physical Systems are increasingly ubiquitous. As sensors and computers become seemingly ever cheaper, many people will entrust their lives to the safe functioning of CPS on a daily basis. Traveling across oceans on a jet-liner or just on a weekly run to the grocery store, the algorithms making critical decisions must choose correctly every time. With this increased dependence on CPS in the core functioning of safety-critical systems, it is more important than ever that these systems be designed and checked in rigorous ways. And in this direction, we've seen many advances in the power of formal verification techniques and in theorem proving for CPS.

Still, many case studies in verification have not held up to industry-sized models. Even worse, in cases where verification is not fully automatic, steep learning curves are a barrier for widespread adoption within industry.

2 Research

Cyber-physical systems can be modeled using *hybrid programs* [3], which capture their discrete behavior with assignments, tests, and loops and model their continuous behavior with differential equations. We use *differential dynamic logic* ($\mathsf{d}\mathcal{L}$) to state and prove properties about hybrid programs.

Using a variant of $\mathsf{d}\mathcal{L}$, we were able to verify safe separation for a class of adaptive cruise controllers for an arbitrary number of cars driving with straight-line dynamics [1]. We set the limits on this class of controllers to be exactly what was necessary to ensure the cars never exited the controllable region. To ensure that our proof extended to all controllers within those limits, we allowed the controller to nondeterministically make any choice within those bounds. By verifying a nondeterministic model, we had the dual benefit that it was easier to verify, since the controller was directly related to the separation property we were proving, and the proof guaranteed safety for a wide

range of implementations. What this work did not include was how to verify that a given controller is in fact within the limits designated by the verified hybrid program, other than checking by hand.

Nondeterministic hybrid programs can also allow a quantified analysis of optimization tradeoffs within a class of controllers which are proven to be safe. For example, in [2], we examined wireless V2V network tradeoffs with the aid of symbolic controllers that are verified safe. We first developed a detailed, symbolic adaptive cruise control, and formally verified the controller is safe for two cars. We then combined probabilistic models of wireless V2V communication with properties of our symbolic controller to find optimal times for making control decisions and/or handing control back to the driver in case no message is received for an extended period of time. However, at the end of the day, it was a series of calculations done by hand that ensured our final result was still a refinement of the originally verified class of controllers.

There are many benefits to verifying nondeterministic hybrid programs, including ease of verification, strength of the verified statement over a range of implementations, and comparison-based analysis of controllers, as mentioned above. But one of the major drawbacks is in their implementation. No engineer would want or allow a finished product to ship while it still exhibits nondeterministic behavior. So, at some point, a deterministic choice must be made. And that choice is ultimately what must be guaranteed safe.

There is currently no formal framework within $\mathsf{d}\mathcal{L}$ to connect a nondeterministic hybrid program which has been verified safe, to a deterministic one without repeating numerous and sometimes costly proof steps. We propose the addition of a series of sound proof rules to the proof calculus for $\mathsf{d}\mathcal{L}$ that will make refinement proofs possible in fewer and cheaper proof steps. By adding such proof rules to $\mathsf{d}\mathcal{L}$, we have the potential to increase the automation of proofs for complicated systems and improve the reusability of hybrid programs which have already been verified. We may also be able to extend verification more easily to legacy systems which are often less accessible for verification than systems which are based on verifiable models in their design.

3 Foundations of CPS in Education

While research and industry motivations for bridging the gap between implementability and ease of verification for cyber-physical systems has already been addressed in Sect. 1 and Sect. 2, there are also several benefits in pedagogical applications of CPS. The course *Foundations of Cyber-Physical Systems* is a newly developed course at CMU, taught for the first time in Fall 2013 by André Platzer [4], with course assignments and labs developed by Sarah Loos. Course enrollment consisted primarily of advanced undergraduates majoring in computer science, and as such, they were very comfortable with imperative programming and had a reasonable familiarity with differential equations. For the majority of students, however, developing models for hybrids systems was still an entirely new area.

Each of the practical lab exercises for this course required students to design a discrete controller, continuous model, and safety specifications for robots to perform a given task. Students were also required to submit a proof that their robots satisfied

the safety specifications. The design of these labs required a careful balance between challenges in proving and challenges in modeling. For example, in the first lab, students are required to model a robot decelerating to come to a stop at a charging station. Since the robots are only allowed to choose a single deceleration, there is a single formula which will yield the correct braking to achieve the goal. Of course, a few students tried to play a guess-and-check game with their submissions, giving specific values for braking, rather than a universal formula. They quickly found that this was always a losing strategy, since the verification step only has a chance of working if their solutions succeed in every case.

In the final lab, students designed their robots to move freely in two-dimensional space, and to avoid static and dynamic obstacles. To verify safe separation between their robots and obstacles, students used nondeterministic programs to capture a wide range of control choices and robot behaviors. Still, many students expressed a desire to create and verify controllers for their robots which optimized certain tasks (fuel consumption or distance from obstacles, for example). However, a particular refinement of their model which optimizes fuel efficiency may entirely obfuscate any connections that controller originally had to the safe separation specifications, taking the verification challenge out of scope at present.

By bridging this gap between hybrid programs that are easy to implement and those that are easy to verify, students can capture a broader understanding of what it means to design and optimize safe CPS. Not only can we make teaching safe CPS easier, but we also stand to increase the usability and power of such verification engines within industry.

References

1. Loos, S.M., Platzer, A., Nistor, L.: Adaptive cruise control: Hybrid, distributed, and now formally verified. In: Butler, M., Schulte, W. (eds.) FM. LNCS, vol. 6664, pp. 42–56. Springer (2011)
2. Loos, S.M., Witmer, D., Steenkiste, P., Platzer, A.: Efficiency analysis of formally verified adaptive cruise controllers. In: Hegyi, A., Schutter, B.D. (eds.) ITSC. pp. 1565–1570. Springer (2013)
3. Platzer, A.: Differential dynamic logic for hybrid systems. J. Autom. Reas. 41(2), 143–189 (2008)
4. Platzer, A.: Teaching CPS foundations with contracts. In: CPS-Ed. pp. 7–10 (2013)

Author Bio. Sarah M. Loos is a Ph.D. student in the Computer Science Department at Carnegie Mellon University. Her research interests include logical analysis and formal verification of distributed hybrid systems, such as distributed car control and collision avoidance protocols for aircraft. She is currently a Department of Energy Computational Science Graduate Fellow and was awarded an NSF Graduate Research Fellowship in 2009. In addition to her role as co-editor of the ACM-W newsletter, Sarah serves as a student member on the board of trustees for the Anita Borg Institute.