The Logical Path to Autonomous Cyber-Physical Systems*
(Invited Paper)

André Platzer[1]

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

Abstract. Autonomous cyber-physical systems are systems that combine the physics of motion with advanced cyber algorithms to act on their own without close human supervision. The present consensus is that reasonable levels of autonomy, such as for self-driving cars or autonomous drones, can only be reached with the help of artificial intelligence and machine learning algorithms that cope with the uncertainties of the real world. That makes safety assurance even more challenging than it already is in cyber-physical systems (CPSs) with classically programmed control, precisely because AI techniques are lauded for their flexibility in handling unpredictable situations, but are themselves harder to predict. This paper identifies the logical path toward autonomous cyber-physical systems in multiple steps. First, differential dynamic logic (dL) provides a logical foundation for developing cyber-physical system models with the mathematical rigor that their safety-critical nature demands. Then, its ModelPlex technique provides a logically correct way to tame the subtle relationship of CPS models to CPS implementations. Finally, the resulting logical monitor conditions can then be exploited to safeguard the decisions of learning agents, guide the optimization of learning processes, and resolve the nondeterminism frequently found in verification models. Overall, logic leads the way in combining the best of both worlds: the strong predictions that formal verification techniques provide alongside the strong flexibility that the use of AI provides.

Keywords: Autonomous cyber-physical systems · Safe AI · Hybrid systems · Differential dynamic logic · Formal verification · Runtime verification

1 Introduction

Autonomous cyber-physical systems (autonomous CPS) use sophisticated software to control the physics of motion. They plan their own goals and actions in pursuit of them. And if things go wrong, they react to situation changes in order

* This material is based upon work supported by the Alexander von Humboldt Foundation, National Science Foundation under NSF CAREER Award CNS-1054246 and CNS-1446712, and US Air Force and DARPA under Contract No. FA8750-18-C-0092.

D. Parker and V. Wolf (Eds.): QEST 2019, LNCS 11785, pp. 25-33, 2019.
DOI: 10.1007/978-3-030-30281-8_2

to prevent problems on their own without close human supervision. Autonomous cyber-physical systems are a technological dream come true. Or are they?

Well, for one thing, cyber-physical systems have found frequent use, but are not yet operated very autonomously. Certainly, the prospects that autonomous cyber-physical systems promise are very appealing, but it is precisely their goal of autonomy and lack of human supervision that also makes them fairly challenging to build just right. Granted, it is also challenging to design an ordinary CPS with human supervision because humans need sufficiently early warning to gain situational awareness and react, which, in turn, requires ample foresight in the CPS design. But the desire for autonomy changes the state of affairs considerably.

From a performance perspective, the biggest difference compared to ordinary CPS is that autonomous CPS do not need to be monitored all the time, but "do the right thing" on their own. The biggest difference from a safety perspective is that it's not clear what the right thing is and humans cannot save the day if the autonomous CPS goes awry, because the whole point is that they are not supervised closely. Autonomy benefits from the help of artificial intelligence and machine learning algorithms that cope with the uncertainties of the real world [27]. Of course, the added flexibility in handling unpredictable situations makes the safety impact of the addition of AI themselves harder to predict.

Formal methods provide ways of establishing safety properties for ordinary CPS [2, 17, 19, 26, 28, 34, 36, 40], and AI provides ways of giving autonomy to CPS. This calls for a combination of formal methods and artificial intelligence [1,9,13,14,16], just not by a friendly ignorance of one another. Instead, the trick is to combine both in a way that each field actually retains its benefits for the CPS in the end. This paper surveys an approach for Safe AI in CPS in which logic leads the way in combining the best of both worlds.

2 Challenge

Cyber-Physical Systems combine cyber capabilities such as communication, computation and control with physical capabilities such as the motion of robots, cars, or aircraft. Mathematical models for such CPS are based on hybrid systems, which combine discrete dynamical systems with continuous dynamical systems, e.g., because discrete change one step at a time fits well to computation, while continuous dynamics along differential equations fits well to their motion.

Formal Verification uses the descriptive models of hybrid systems for predicting, with the help of model checking [8, 11] or proof [31, 36], whether all their behavior satisfies safety properties of interest, such as collision freedom. Especially in the case of logical proofs, formal methods enable very strong guarantees about *all* behavior of the mathematical models from a small reasoning basis [35]. In order to overcome complexity challenges, it is often important to work with models that use simplifying abstractions, because models that include literally all implementation detail quickly become prohibitively expensive to analyze.

Machine Learning forgoes the principle of explicitly programming all behavior of a system and, instead, uses learning algorithms that generalize responses from static data (e.g., a set of labeled images to classify) or from dynamic experience (e.g., responses to trial and error). *Reinforcement learning* (RL) [39], for example, repeatedly tries out an action, observes what the overall outcome of a sequence of actions was, and then increases the probability with which its policy decides for actions that have had large fractions of good outcomes so far. The big advantage of reinforcement learning is that it can be used with very minimal assumptions on the system to be controlled. All it takes is a black-box way of executing actions and reliably observing the outcome, e.g., in a simulator. In practice, learning systems are also lauded for their flexibility in responding to situations that were not directly programmed into the system design. Learning is, thus, presently considered crucial to reach reasonable levels of autonomy.

Of course, guarantees are harder to come by. At the very least, one has to assume that the outcomes observed for the individual actions in the individual states are strongly correlated (in fact, Markovian) with outcomes at other times. Under suitable assumptions, finite-state cases provide elegant theoretical guarantees [39]. But the infinite-state case of CPSs is significantly more complex, because even the luxury of an arbitrary countable amount of experiments is not enough to try all actions in all states. Indeed, black-box uses require fairly strong additional assumptions to enable any correct predictions at all [6, 37, 41], and many of those assumptions need to be provided as explicit inputs into safety analysis algorithms for soundness. In particular, a white-box model is required to obtain guarantees even if only an executable model is needed during learning.

Safety for Autonomous CPS requires direct attention to the interplay of learning systems with hybrid system models. Even if the combination of learning algorithms with the CPS dynamics formally are hybrid systems again, they cannot be considered quite as naïvely due to the resulting scale. Without summarizing symbolic abstractions, it would have been completely infeasible, for example, to verify the hybrid systems model of the next-generation Airborne Collision Avoidance System ACAS X [18] defined by interpolation of a Markov Decision Process policy on its half a trillion different discretized state regions. Instead, the computational complexities call for approaches that establish safety from simpler models that do not include full detail on the learning while still benefiting from the flexibility advantages of learning without risking unsafety.

3 Approach

As a foundation for the safe design of autonomous CPS, this approach uses *differential dynamic logic* dL [30, 31, 33, 36] which provides modalities $[\alpha]$ and $\langle\alpha\rangle$ for every hybrid system model α such that the dL formula $[\alpha]\phi$ is true in a state whenever the postcondition ϕ is true after all runs of α (safety) and the dL formula $\langle\alpha\rangle\phi$ is true in a state whenever ϕ is true after at least one run of

α (liveness). Besides serving as a flexible specification language, dL also comes with axiomatizations [30, 32, 33, 35] that enable its use for verification purposes.

CPS Modeling is the first step and culminates in a hybrid system α describing all possible behavior of the CPS. For both complexity control reasons and flexibility reasons, it is best *not* to describe completely accurately under which exact circumstance the learning system decides upon which exact control action. Instead, the hybrid system α describes all actions that are possible as well as the continuous dynamics of the system.

Elaborate modeling advice can be found elsewhere [36,38], but nondeterminism is frequently used for this purpose. For example, a hybrid systems model

$$\big((\beta \cup \gamma); x' = f(x)\big)^* \tag{1}$$

expresses that the CPS can nondeterministically choose (by operator $\cup$) to either run control action β or control action γ and will then (after the ; operator) follow the continuous dynamics of the differential equation $x' = f(x)$ for a certain period of time, before repeating (by operator * for repetition) the sequence of discrete and continuous actions any number of times. For example, β could be the action of accelerating while γ could be braking (additional actions such as turning left add more $\cup$ operators, accordingly). A model of this shape is fairly noncommittal, because its use of nondeterminism in action choices, differential equation durations, and repetition counts deliberately leaves open how exactly it is run, giving the learning CPS a lot of flexibility in filling in these choices at its leisure later without requiring any change in the model.

KeYmaera X: Hybrid Systems Model Safety can be established by proving in the tool KeYmaera X [12] a dL safety property of the form

$$\phi \to [\alpha]\psi \tag{2}$$

which, if proved, implies that, if the system starts in any initial state satisfying formula ϕ, then all states reached after all runs of the hybrid systems model α satisfy formula ψ. Formal proofs of dL formulas such as (2) are highly trustworthy, not just because of the clever design of KeYmaera X that reduces its soundness-critical core to less than 2000 lines of code [12] but also because of the cross-verification of the soundness of dL in both Isabelle/HOL and Coq [3].

A formal proof of (2) justifies that all behavior of α satisfies the safety property. The most valuable takeaway lesson besides the formal proof itself are the additional requirements inevitably found during the proof, which characterize when it is even safe to use the various control actions in the model α. For example, in the initial model (1), actions β and γ were unconstrained, but it may not always be safe to accelerate without first checking a condition C that, e.g., relates the obstacle distance to the present velocities and braking capabilities:

$$\big((?C; \beta) \cup \gamma); x' = f(x)\big)^* \tag{3}$$

This refined hybrid system (3) includes an additional test (written $?C$) that needs to pass since C holds true before running action β. If C is true in the present state, then both β and γ can be run by a nondeterministic choice ($\cup$), otherwise only γ is available, because the condition $?C$ would fail. *All* such additional constraints that are required for safety will be discovered during the proof of (2), because a sound proof could not otherwise succeed, and dL is sound [30, 32, 35].

ModelPlex: Model Safety Transfer provides the correctness bridge between a verified hybrid systems model and its implementation by synthesizing correct-by-construction runtime monitors. A dL proof of formula (2) in KeYmaera X is a great achievement, but, due to its (desirable) modeling simplifications, does not provide an answer for the full complexities of a learning CPS. Usually, there is a discrepancy between the implementation detail of the autonomous CPS and the simplified descriptions that were chosen to be included in the verified model. Fortunately, the ModelPlex procedure [24] can overcome such discrepancies. Given a verified dL model, ModelPlex synthesizes a monitor along with a dL correctness proof for it, saying that the real implementation is safe as long as it satisfies that runtime monitor (and will always remain safe when continuing the model).

The same relationship between verified model and runtime monitor also is the cornerstone to safeguard the decisions of learning agents [14], which is crucial to obtain safety after deployment unless ModelPlex has already been used during learning to guide the learning process toward safe answers (which speeds up convergence). The logical monitor conditions obtained from a ModelPlex proof can be directly exploited as a safety signal for learning. Since it is challenging to implement learning algorithms in a provably correct way, the continued use of ModelPlex monitors after deployment is advisable even if ModelPlex monitor outputs were used to steer learning toward safe answers during training.

VeriPhy: Executable Proof Transfer synthesizes executable machine code binaries (e.g., for x64 or ARM) that inherit the safety theorems such as (2) by a chain of formal proofs in theorem provers [4]. The resulting executables are not just formally verified to be safe for the CPS, but also accept control input from unverified controllers that will be checked against ModelPlex monitors for safety before execution and are vetoed otherwise. This input decides how to resolve the nondeterminism in the hybrid systems model, e.g., whether to run β or γ in (3). But the verified controller sandbox generated by VeriPhy only accepts β if the condition C was true that was required for the safety proof. While the need to test C when deciding on β was evident from the way model (3) was written, other conditions are more difficult to read off, and the key is to find them all and then prove safety of the control sandbox, which VeriPhy does automatically.

Safe Learning in CPSs is made possible by the combination of a hybrid systems model verified in KeYmaera X [12], whose safety-critical monitor conditions were extracted along with a proof of correctness by ModelPlex [24], and

whose verified controller sandbox was synthesized along with a chain of correctness proofs by VeriPhy [4]. This combination enables any reinforcement learning algorithm to be run as a black box [14]. The VeriPhy output provides a verified CPS sandbox within which the reinforcement learning can experiment safely. The reinforcement learning algorithm can focus on identifying the most optimal decisions, which is usually replaced by nondeterminism in verification for the sake of simplicity. Convergence of the learning algorithm is improved, because the ModelPlex monitors give immediate feedback about which individual action might cause an unsafe future in which state. This is faster than having to wait until an entire sequence of actions has been chosen that, say, lead to a collision, and then facing the nontrivial task of retroactively identifying to what extent which action contributed to this collision and back-propagate generalizable knowledge.

If the physical behavior was modeled adequately, then this approach leads to a provably safe policy [14]. Otherwise, quantitative ModelPlex, which gives a real-valued (instead of boolean-valued) degree of compliance, is experimentally shown to guide the optimization of reinforcement learning (RL) off model to a graceful recovery using the ability of boolean ModelPlex to reliably spot when the real behavior is outside the verified model. The question is what could then prove safety regardless, not just observe recovery. Clearly, if all model assumptions are completely wrong, then no amount of analysis will make the system safe but magic is needed instead. Yet, if there merely is uncertainty about which one of a whole pile of models is the right one, yet they are not all wrong, then not only is safety preserved, but learning can also optimize the system by actively experimenting to find out which model accurately reflects the present reality [15]. Conjunctions of the ModelPlex monitors for all plausible models keep the learning AI safe. Solving for distinct monitor predictions makes it possible to plan differentiating experiments to converge a.s. to the true model, if possible. When the verified models are given together with a tactic that proves them, then safety proofs can be reified, such that both the model and its safety proof can be adapted to better fit observations with *verification-preserving model updates*.

4 Summary and Outlook

Overall, logic leads the way in combining the best of both worlds: the strong predictions that formal verification techniques provide for CPS alongside the strong flexibility that the use of AI provides. Table 1 summarizes the logical technolo-

Table 1. Logical triumvirate of technologies for transitioning trustworthiness to autonomous cyber-physical systems

dL Proof Technology	**RL Learning Technology**
KeYmaera X: identify safe actions in CPS model	RL optimizes action choice
ModelPlex: safe model to safe implementation	safe reward signal for RL
VeriPhy: monitored sandbox to safe executables	CPS sandbox for RL

gies that enable the respective combinations, their uses in CPS design, and their corresponding counterpart in AI. Each element of the safety transition stack fulfills a different purpose and integrates the benefits of learning and proving in different ways. They all share differential dynamic logic dL as a common logical foundation and reinforcement learning RL as a learning foundation.

While logic paints a particularly clear picture of how to safely navigate the path to autonomous CPSs, and while its efficacy has been demonstrated throughout on small scale [5], numerous interesting challenges remain that go beyond the ones of interest already for ordinary CPS [34]. The guarantees, even in the presence of learning, are strong on the controls side of CPS. The safety-relevant control error is provably reduced to zero thanks to the logical safety stack, but only under the assumption of bounded deviations in sensing [24].

The picture is not so rosy on the sensing side of CPS. And I argue that this is not a coincidence. Of course, no amount of reasoning can bypass the sensory illusions of the Cartesian Demon that fooled all but René Descartes' existence of thoughts [7]. If literally *all* sensors and actuators of a CPS could be arbitrarily wrong, then no connection can be made between the suspected and real state of the system. But even if sensors are almost always a little wrong, they are not usually all that wrong, which enables a logical angle of attack [21, 23, 25] for guarantees despite bounded sensor errors. Now, how can concrete bounds be substantiated for sensor errors with as little doubt as possible? An answer to this question is the true challenge beyond recent progress in verified perception [10, 29].

References

1. Alshiekh, M., Bloem, R., Ehlers, R., Könighofer, B., Niekum, S., Topcu, U.: Safe reinforcement learning via shielding. In: McIlraith and Weinberger [22]
2. Alur, R.: Formal verification of hybrid systems. In: Chakraborty, S., Jerraya, A., Baruah, S.K., Fischmeister, S. (eds.) EMSOFT. pp. 273–278. ACM, New York (2011). doi: 10.1145/2038642.2038685
3. Bohrer, B., Rahli, V., Vukotic, I., Völp, M., Platzer, A.: Formally verified differential dynamic logic. In: Bertot, Y., Vafeiadis, V. (eds.) Certified Programs and Proofs - 6th ACM SIGPLAN Conference, CPP 2017, Paris, France, January 16-17, 2017. pp. 208–221. ACM, New York (2017). doi: 10.1145/3018610.3018616
4. Bohrer, B., Tan, Y.K., Mitsch, S., Myreen, M.O., Platzer, A.: VeriPhy: Verified controller executables from verified cyber-physical system models. In: Grossman, D. (ed.) Proceedings of the 39th ACM SIGPLAN Conference on Programming Language Design and Implementation, PLDI 2018. pp. 617–630. ACM (2018). doi: 10.1145/3192366.3192406
5. Bohrer, B., Tan, Y.K., Mitsch, S., Sogokon, A., Platzer, A.: A formal safety net for waypoint following in ground robots. IEEE Robotics and Automation Letters . doi: 10.1109/LRA.2019.2923099, to appear
6. Collins, P.: Optimal semicomputable approximations to reachable and invariant sets. Theory Comput. Syst. **41**(1), 33–48 (2007). doi: 10.1007/s00224-006-1338-3
7. Descartes, R.: Meditationes de prima philosophia, in qua Dei existentia et animae immortalitas demonstratur (1641)

8. Doyen, L., Frehse, G., Pappas, G.J., Platzer, A.: Verification of hybrid systems. In: Clarke, E.M., Henzinger, T.A., Veith, H., Bloem, R. (eds.) Handbook of Model Checking, chap. 30, pp. 1047–1110. Springer (2018). doi: 10.1007/978-3-319-10575-8_30
9. Dreossi, T., Donzé, A., Seshia, S.A.: Compositional falsification of cyber-physical systems with machine learning components. In: Barrett, C., Davies, M., Kahsai, T. (eds.) NASA Formal Methods - 9th International Symposium, NFM 2017, Moffett Field, CA, USA, May 16-18, 2017, Proceedings. LNCS, vol. 10227, pp. 357–372 (2017). doi: 10.1007/978-3-319-57288-8_26
10. Dvijotham, K., Gowal, S., Stanforth, R., Arandjelovic, R., O'Donoghue, B., Uesato, J., Kohli, P.: Training verified learners with learned verifiers. CoRR **abs/1805.10265** (2018)
11. Frehse, G., Guernic, C.L., Donzé, A., Cotton, S., Ray, R., Lebeltel, O., Ripado, R., Girard, A., Dang, T., Maler, O.: SpaceEx: Scalable verification of hybrid systems. In: Gopalakrishnan, G., Qadeer, S. (eds.) CAV. LNCS, vol. 6806, pp. 379–395. Springer, Berlin (2011). doi: 10.1007/978-3-642-22110-1_30
12. Fulton, N., Mitsch, S., Quesel, J.D., Völp, M., Platzer, A.: KeYmaera X: An axiomatic tactical theorem prover for hybrid systems. In: Felty, A., Middeldorp, A. (eds.) CADE. LNCS, vol. 9195, pp. 527–538. Springer, Berlin (2015). doi: 10.1007/978-3-319-21401-6_36
13. Fulton, N., Platzer, A.: Safe AI for CPS. In: IEEE International Test Conference, ITC 2018, Phoenix, AZ, USA, October 29 - Nov. 1, 2018. pp. 1–7. IEEE (2018). doi: 10.1109/TEST.2018.8624774
14. Fulton, N., Platzer, A.: Safe reinforcement learning via formal methods: Toward safe control through proof and learning. In: McIlraith and Weinberger [22], pp. 6485–6492, https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/17376
15. Fulton, N., Platzer, A.: Verifiably safe off-model reinforcement learning. In: Vojnar, T., Zhang, L. (eds.) TACAS, Part I. LNCS, vol. 11427, pp. 413–430. Springer (2019). doi: 10.1007/978-3-030-17462-0_28
16. Gillula, J.H., Tomlin, C.J.: Guaranteed safe online learning via reachability: tracking a ground target using a quadrotor. In: IEEE International Conference on Robotics and Automation, ICRA 2012, 14-18 May, 2012, St. Paul, Minnesota, USA. pp. 2723–2730. IEEE (2012). doi: 10.1109/ICRA.2012.6225136
17. Henzinger, T.A., Sifakis, J.: The discipline of embedded systems design. Computer **40**(10), 32–40 (10 2007). doi: 10.1109/MC.2007.364
18. Jeannin, J., Ghorbal, K., Kouskoulas, Y., Schmidt, A., Gardner, R., Mitsch, S., Platzer, A.: A formally verified hybrid system for safe advisories in the next-generation airborne collision avoidance system. STTT **19**(6), 717–741 (2017). doi: 10.1007/s10009-016-0434-1
19. Larsen, K.G.: Verification and performance analysis for embedded systems. In: Chin, W., Qin, S. (eds.) TASE 2009, Third IEEE International Symposium on Theoretical Aspects of Software Engineering, 29-31 July 2009, Tianjin, China. pp. 3–4. IEEE Computer Society (2009). doi: 10.1109/TASE.2009.66
20. Logic in Computer Science (LICS), 2012 27th Annual IEEE Symposium on. IEEE, Los Alamitos (2012)
21. Martins, J., Platzer, A., Leite, J.: Dynamic doxastic differential dynamic logic for belief-aware cyber-physical systems. In: Cerrito, S., Popescu, A. (eds.) TABLEAUX. LNCS, Springer (2019)

22. McIlraith, S.A., Weinberger, K.Q. (eds.): Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, Louisiana, USA, February 2-7, 2018. AAAI Press (2018)
23. Mitsch, S., Ghorbal, K., Vogelbacher, D., Platzer, A.: Formal verification of obstacle avoidance and navigation of ground robots. I. J. Robotics Res. **36**(12), 1312–1340 (2017). doi: 10.1177/0278364917733549
24. Mitsch, S., Platzer, A.: ModelPlex: Verified runtime validation of verified cyber-physical system models. Form. Methods Syst. Des. **49**(1-2), 33–74 (2016). doi: 10.1007/s10703-016-0241-z, special issue of selected papers from RV'14
25. Mitsch, S., Platzer, A.: Verified runtime validation for partially observable hybrid systems. CoRR **abs/1811.06502** (2018), http://arxiv.org/abs/1811.06502
26. Nerode, A.: Logic and control. In: Cooper, S.B., Löwe, B., Sorbi, A. (eds.) CiE. LNCS, vol. 4497, pp. 585–597. Springer, Berlin (2007). doi: 10.1007/978-3-540-73001-9_61
27. Paden, B., Cáp, M., Yong, S.Z., Yershov, D.S., Frazzoli, E.: A survey of motion planning and control techniques for self-driving urban vehicles. IEEE Trans. Intelligent Vehicles **1**(1), 33–55 (2016). doi: 10.1109/TIV.2016.2578706
28. Pappas, G.J.: Wireless control networks: modeling, synthesis, robustness, security. In: Caccamo, M., Frazzoli, E., Grosu, R. (eds.) HSCC. pp. 1–2. ACM, New York (2011). doi: 10.1145/1967701.1967703
29. Pei, K., Cao, Y., Yang, J., Jana, S.: Towards practical verification of machine learning: The case of computer vision systems. CoRR **abs/1712.01785** (2017)
30. Platzer, A.: Differential dynamic logic for hybrid systems. J. Autom. Reas. **41**(2), 143–189 (2008). doi: 10.1007/s10817-008-9103-8
31. Platzer, A.: Logical Analysis of Hybrid Systems: Proving Theorems for Complex Dynamics. Springer, Heidelberg (2010). doi: 10.1007/978-3-642-14509-4
32. Platzer, A.: The complete proof theory of hybrid systems. In: LICS [20], pp. 541–550. doi: 10.1109/LICS.2012.64
33. Platzer, A.: Logics of dynamical systems. In: LICS [20], pp. 13–24. doi: 10.1109/LICS.2012.13
34. Platzer, A.: Logic & proofs for cyber-physical systems. In: Olivetti, N., Tiwari, A. (eds.) IJCAR. LNCS, vol. 9706, pp. 15–21. Springer, Cham (2016). doi: 10.1007/978-3-319-40229-1_3
35. Platzer, A.: A complete uniform substitution calculus for differential dynamic logic. J. Autom. Reas. **59**(2), 219–265 (2017). doi: 10.1007/s10817-016-9385-1
36. Platzer, A.: Logical Foundations of Cyber-Physical Systems. Springer, Cham (2018). doi: 10.1007/978-3-319-63588-0
37. Platzer, A., Clarke, E.M.: The image computation problem in hybrid systems model checking. In: Bemporad, A., Bicchi, A., Buttazzo, G.C. (eds.) HSCC. LNCS, vol. 4416, pp. 473–486. Springer (2007). doi: 10.1007/978-3-540-71493-4_37
38. Quesel, J.D., Mitsch, S., Loos, S., Aréchiga, N., Platzer, A.: How to model and prove hybrid systems with KeYmaera: A tutorial on safety. STTT **18**(1), 67–91 (2016). doi: 10.1007/s10009-015-0367-0
39. Sutton, R.S., Barto, A.G.: Reinforcement Learning. The MIT Press, Cambridge (1998)
40. Tiwari, A.: Logic in software, dynamical and biological systems. In: LICS. pp. 9–10. IEEE Computer Society (2011). doi: 10.1109/LICS.2011.20
41. Zuliani, P., Platzer, A., Clarke, E.M.: Bayesian statistical model checking with application to Simulink/Stateflow verification. Form. Methods Syst. Des. **43**(2), 338–367 (2013). doi: 10.1007/s10703-013-0195-3