15-424: Foundations of Cyber-Physical Systems

Lecture Notes on Dynamical Systems & Dynamic Axioms

André Platzer

Carnegie Mellon University
Lecture 5

1 Introduction

Lecture 4 on Safety & Contracts demonstrated how useful and crucial CPS contracts are for CPS. Their role and understanding goes beyond dynamic testing. In CPS, proven CPS contracts are infinitely more valuable than dynamically tested contracts, because dynamical tests of contracts at runtime of a CPS generally leave open very little flexibility for reacting to them in any safe way. After all, the failure of a contract indicates that some safety condition that was expected to hold is not longer true. Unless provably sufficient safety margins and fallback plans remain, the system is already in trouble then.[1]

Consequently, CPS contracts really shine in relation to how they are proved for CPS. Understanding how to prove CPS contracts requires us to understand the dynamical effects of hybrid programs in more detail. This deeper understanding of the effects of hybrid program statements is not only useful for conducting proofs but also for developing and sharpening our intuition about hybrid programs for CPS. This phenomenon illustrates a more general point that proof and effect (and/or meaning) are intimately linked and that truly understanding effect is ultimately the same as, as well as a prerequisite to, understanding how to prove properties of that effect [Pla12d, Pla12b, Pla10]. You may have seen this point demonstrated already in other courses from the Principles of Programming Languages group at CMU, but it will shine in today's lecture.

The route that we choose to get to this level of understanding is one that involves a closer look at dynamical systems and Kripke models, or rather, the effect that hybrid

[1] Although, in combination with formal verification, the Simplex architecture can be understood as exploiting the relationship of dynamic contracts for safety purposes [SKSC98]. ModelPlex, which is based on differential dynamic logic, lifts this observation to a fully verified link from verified models to CPS executions [MP16].

programs have on them. This will enable us to devise authoritative proof principles for differential dynamic logic and hybrid programs [Pla12d, Pla12b, Pla10, Pla08]. While there are many more interesting things to say about dynamical systems and Kripke structures, this lecture will limit information to the truly essential parts that are crucial right now and leave more elaboration for later lectures. Today's lecture will give us the essential reasoning tools for cyber-physical systems and is, thus, of central importance.

More information can be found in [Pla12c, Pla12d] as well as [Pla10, Chapter 2.3].

The focus of today's lecture is on a systematic development of the basic reasoning principles for cyber-physical systems. The goal is to cover all cyber-physical systems by identifying one fundamental reasoning principle for each of the operators of differential dynamic logic and, specifically, its hybrid programs. Once we have a (suitably complete) reasoning principle for each of the operators, the basic idea is that any arbitrary cyber-physical system can be analyzed by just combining the various reasoning principles with one another, compositionally, by inspecting one operator at a time.

Note 1 (Logical guiding principle: Compositionality)**.** *Since every CPS is modeled by a hybrid program*[a] *and all hybrid programs are combinations of simpler hybrid programs by one of a handful of program operators (such as* $\cup$ *and* ; *and* **), all CPS can be analyzed if only we identify one suitable analysis technique for each of the operators.*

[a] To faithfully represent complex CPS, some models need an extension of hybrid programs, e.g., to hybrid games [Pla15a] or distributed hybrid programs [Pla12a], in which case suitable generalizations of the logical approach presented here works.

With enough understanding, this guiding principle will, indeed, ultimately succeed [Pla12b, Pla14]. It will, however, take significantly more than one lecture to get there. So, today's lecture will settle for a systematic development of the reasoning principles for the more elementary operators in hybrid programs, leaving a detailed development of the others to later lectures.

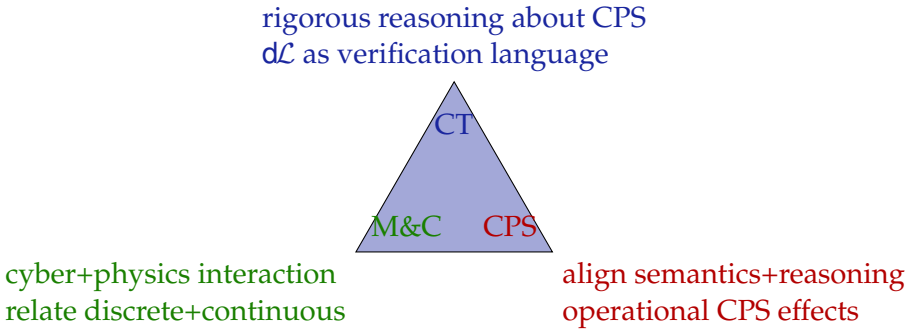
This lecture is of central significance for the Foundations of Cyber-Physical Systems. The most important learning goals of this lecture are:

Modeling and Control: We will understand the core principles behind CPS by understanding analytically and semantically how cyber and physical aspects are integrated and interact in CPS. This lecture will also begin to explicitly relate discrete and continuous systems, which will ultimately lead to a fascinating view on understanding hybridness [Pla12b].

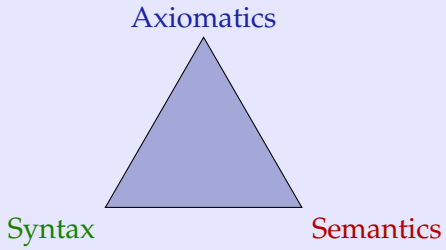
Computational Thinking: This lecture is devoted to the core aspects of reasoning rigorously about CPS models, which is critical to getting CPS right. CPS designs can be flawed for very subtle reasons. Without sufficient rigor in their analysis it can be impossible to spot the flaws, and even more challenging to say for sure whether and why a design is no longer faulty. This lecture systematically develops one reasoning principle for each of the operators of hybrid programs. This

lecture begins an *axiomatization* of differential dynamic logic dℒ [Pla12d, Pla12b] to lift dℒ from a specification language to a verification language for CPS.

CPS Skills: We will develop a deep understanding of the semantics of CPS models by carefully relating their semantics to their reasoning principles and aligning them in perfect unison. This understanding will also enable us to develop a better intuition for the operational effects involved in CPS.



Note 2 (Logical trinity). *The concepts developed in this lecture illustrate the more general relation of* syntax *(which is notation),* semantics *(what carries meaning), and* axiomatics *(which internalizes semantic relations into universal syntactic transformations). These concepts and their relations jointly form the significant* logical trinity *of syntax, semantics, and axiomatics.*



2 Intermediate Conditions for CPS

Recall the bouncing ball from Lecture 4 on Safety & Contracts

$$0 \leq x \wedge x = H \wedge v = 0 \wedge g > 0 \wedge 1 > c \geq 0 \rightarrow \\ \left[\left(x' = v, v' = -g \,\&\, x \geq 0;\ (?x = 0; v := -cv \cup ?x \neq 0)\right)^*\right](0 \leq x \wedge x \leq H) \quad (1)$$

To simplify the subsequent discussion, let's drop the repetition (*) for now:

$$0 \leq x \wedge x = H \wedge v = 0 \wedge g > 0 \wedge 1 \geq c \geq 0 \rightarrow \\ \left[x' = v, v' = -g \,\&\, x \geq 0; (?x = 0; v := -cv \cup ?x \neq 0)\right](0 \leq x \wedge x \leq H) \quad (2)$$

Of course, dropping the repetition grotesquely changes the behavior of the bouncing ball. It cannot even really bounce any longer now. It can merely fall and revert its velocity vector when on the ground but is then stuck. The single hop bouncing ball can only follow the first blue hop but not the gray remainder hops in Fig. 1. This degenerate model fragment is, nevertheless, an insightful stepping stone toward a proof of the full model. If we manage to prove (2), we certainly have not shown the full bouncing ball

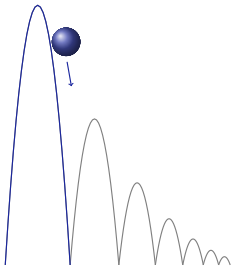


Figure 1: Sample trajectory of a single-hop bouncing ball (plotted as height over time) which can follow the first blue hop but is incapable of following the remainder shown in gray.

formula (1). But it's a start, because the behavior modeled in (2) is a part of the behavior of (1). So it is useful (and easier) to understand (2) first.

The dL formula (2) has a number of assumptions $0 \leq x \wedge x = H \wedge v = 0 \wedge g > 0 \wedge 1 \geq c \geq 0$ that can be used during the proof. It claims that the postcondition $0 \leq x \wedge x \leq H$ holds after all runs of the HP in the $[\cdot]$ modality. The top-level operator in the modality of (2) is a sequential composition (;), for which we need to find a proof argument.[2]

The HP in (2) follows a differential equation first and then, after the sequential composition (;), proceeds to run a discrete program $(?x = 0; v := -cv \cup ?x \neq 0)$. Depending on how long the HP follows its differential equation, the intermediate state after the differential equation and before the discrete program will be rather different.

Note 3 (Intermediate states of sequential compositions)**.** *This phenomenon happens in general for sequential compositions $\alpha;\beta$. The first HP α may reach a whole range of states, which represent intermediate states for the sequential composition $\alpha;\beta$, i.e. states that are final states for α and initial states for β. The intermediate states of $\alpha;\beta$ are the states μ in the semantics $[\![\alpha;\beta]\!]$ from Lecture 3:*

$$[\![\alpha;\beta]\!] = [\![\alpha]\!] \circ [\![\beta]\!] = \{(\omega,\nu) : (\omega,\mu) \in [\![\alpha]\!], (\mu,\nu) \in [\![\beta]\!]\}$$

It turns out we can summarize what all intermediate states between the differential equation and the discrete program of (2) have in common. They differ by how long

[2] The way we proceed here to prove (2) is actually not the recommended way. Later on, we will see a much easier way. But it is instructive to understand the more verbose approach we take first. The approach we follow first also prepares us for the challenges that lie ahead when proving properties of loops.

the CPS has followed the differential equation. But the intermediate states still have in common that they satisfy a logical formula E. Which logical formula that is, is, in fact, instructive to find out, but of no immediate concern for the rest of this lecture. So we invite you to find out how to choose E for (2) before you compare your answer to the one we develop in the appendix of Lecture 4 on Safety & Contracts.

For a HP that is a sequential composition $\alpha;\beta$ an *intermediate condition* is a formula that characterizes the intermediate states in between HP α and β. That is, for a dL formula

$$A \to [\alpha;\beta]B$$

an intermediate condition is a formula E such that the following dL formulas are valid:

$$A \to [\alpha]E \qquad \text{and} \qquad E \to [\beta]B$$

The first dL formula expresses that intermediate condition E characterizes the intermediate states accurately, i.e. E actually holds after all runs of HP α from states satisfying A. The second dL formula says that the intermediate condition E characterizes intermediate states well enough, i.e. E is all we need to know about a state to conclude that all runs of β end up in B. That is, from all states satisfying E (in particular from those that result by running α from a state satisfying A), B holds after all runs of β.

Intermediate condition contracts for sequential compositions are captured more concisely in the following proof rule:

$$\text{G[;]}\ \frac{A \to [\alpha]E \quad E \to [\beta]B}{A \to [\alpha;\beta]B}$$

The two dL formulas above the bar of the rule are called *premises*. The dL formula below the bar is called *conclusion*. The argument above (still informally) justifies the proof rule because if both premises are valid then the conclusion is valid, too. So if we have a proof for each of the two premises, rule G[;] gave us a proof of the conclusion.

Since we will soon identify a better way of proving properties of sequential compositions, we do not pursue rule G[;] further now. Note, however, that there are some circumstances under which a proof using G[;] actually simplifies your reasoning.

For now, we remark that, given an intermediate condition E, the rule G[;] splits a proof of (2) into a proof of the following two premises:

$$0 \le x \land x = H \land v = 0 \land g > 0 \land 1 \ge c \ge 0 \to \left[x' = v, v' = -g \,\&\, x \ge 0\right] E \tag{3}$$

$$E \to \left[?x = 0; v := -cv \cup ?x \ne 0)\right] (0 \le x \land x \le H) \tag{4}$$

3 Dynamic Axioms for Nondeterministic Choices

By the logical guiding principle of compositionality (Note 1), the next operator that we need to understand in order to prove (2) is the nondeterministic choice $?x = 0; v := -cv \cup$

$?x \neq 0$ in (4). By guiding principle Note 1, we zero in on the nondeterministic choice operator $\cup$ and pretend we already knew how to handle all other operators.

Recall the semantics of nondeterministic choices from Lecture 3 on Choice & Control:

$$[\![\alpha \cup \beta]\!] = [\![\alpha]\!] \cup [\![\beta]\!] \tag{5}$$

Remember that $[\![\alpha]\!]$ is a reachability relation on states, where $(\omega, \nu) \in [\![\alpha]\!]$ iff HP α can run from state ω to state ν. Let us illustrate graphically what (5) means:

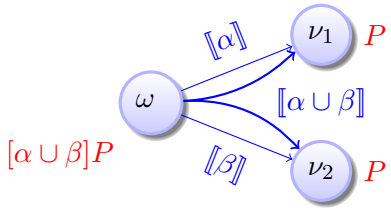


Figure 2: Illustration of the transition semantics of $\alpha \cup \beta$, which allows $\alpha \cup \beta$ to make any transitions that either α or that β could do on their own.

According to $[\![\alpha]\!]$, a number of states ν_i are reachable by running HP α from some initial state ω.[3] According to $[\![\beta]\!]$, a number of (possibly other) states ν_i are reachable by running HP β from the same initial state ω. By the semantic equation (5), running $\alpha \cup \beta$ from ω can give us any of those possible outcomes. And there was nothing special about the initial state ω. The same principle holds for all other states as well.

Note 4 ($\cup$). *The nondeterministic choice $\alpha \cup \beta$ can lead to exactly the states to which either α could take us or to which β could take us or to which both could lead. The dynamic effect of a nondeterministic choice $\alpha \cup \beta$ is that running it at any time either results in a behavior of α or of β, nondeterministically. So both the behaviors of α and β are possible when running $\alpha \cup \beta$.*

If we want to understand whether and where dL formula $[\alpha \cup \beta]P$ is true, we need to understand which states the modality $[\alpha \cup \beta]$ refers to. In which states does P have to be true so that $[\alpha \cup \beta]P$ is true in state ω?

By definition of the semantics, P needs to be true in all states that $\alpha \cup \beta$ can reach according to $[\![\alpha \cup \beta]\!]$ from ω for $[\alpha \cup \beta]P$ to be true in ω. Referring to semantics (5) or looking at Fig. 2, shows us that this includes exactly all states that α can reach from ω according to $[\![\alpha]\!]$, hence $[\alpha]P$ has to be true in ω. And that it also includes all states that β can reach from ω, hence $[\beta]P$ has to be true in ω.

Consequently,

$$\omega \in [\![[\alpha]P]\!] \quad \text{and} \quad \omega \in [\![[\beta]P]\!] \tag{6}$$

are necessary conditions for

$$\omega \in [\![[\alpha \cup \beta]P]\!] \tag{7}$$

[3] Fig. 2 only illustrates one such state ν_1 for visual conciseness. But ν_1 should be thought of as a generic representative for any such state that α can reach from the initial state ω in these figures.

That is, unless (6) holds, (7) cannot possibly hold. So (6) is necessary for (7). Are there any states missing? Are there any states that (7) would require to satisfy P, which (6) does not already ensure to satisfy P? No, because, by (5), $\alpha \cup \beta$ does not admit any behavior that neither α nor β can exhibit. Hence (6) is also sufficient for (7), i.e. (6) implies (7). So (6) and (7) are equivalent.

Thus, when adopting a more logical language again, this justifies:

$$\omega \in [\![[\alpha \cup \beta]P \leftrightarrow [\alpha]P \wedge [\beta]P]\!]$$

This reasoning did not depend on the particular state ω but holds for all ω. Therefore, the formula $[\alpha \cup \beta]P \leftrightarrow [\alpha]P \wedge [\beta]P$ is valid, written:

$$\models [\alpha \cup \beta]P \leftrightarrow [\alpha]P \wedge [\beta]P$$

Exciting! We have just proved our first axiom to be sound:

Lemma 1 ([∪] soundness)**.** *The "axiom of choice" is* sound*, i.e. all its instances are valid:*

$$[\cup] \quad [\alpha \cup \beta]P \leftrightarrow [\alpha]P \wedge [\beta]P$$

Nondeterministic choices split into their alternatives in axiom [∪]. From right to left: If all α runs lead to states satisfying P (i.e., $[\alpha]P$ holds) and all β runs lead to states satisfying P (i.e., $[\beta]P$ holds), then all runs of HP $\alpha \cup \beta$, which may choose between following α and following β, also lead to states satisfying P (i.e., $[\alpha \cup \beta]P$ holds). The converse implication from left to right holds, because $\alpha \cup \beta$ can run all runs of α and all runs of β, so all runs of α (and of β) lead to states satisfying P if that holds for all runs of $[\beta]P$.

Armed with this axiom [∪] at our disposal, we can now easily make the following inference just by invoking the equivalence that [∪] justifies.

$$[\cup]\frac{A \rightarrow [\alpha]B \wedge [\beta]B}{A \rightarrow [\alpha \cup \beta]B}$$

Let's elaborate. If we want to prove the conclusion

$$A \rightarrow [\alpha \cup \beta]B \tag{8}$$

then we can instead prove the premise

$$A \rightarrow [\alpha]B \wedge [\beta]B \tag{9}$$

because by [∪], or rather an instance of [∪] formed by using B for P, we know:

$$[\alpha \cup \beta]B \leftrightarrow [\alpha]B \wedge [\beta]B \tag{10}$$

Since (10) is a valid equivalence, its left-hand side and its right-hand side are equivalent. Wherever the left-hand side occurs, we can equivalently replace it by the right-hand

side, since both are equivalent.[4] Thus, replacing the place where the left-hand side of (10) occurs in (8) by the right-hand side of (10) gives us the formula (9) that is equivalent to (8). After all, according to the valid equivalence (10) justified by axiom [∪], (9) can be obtained from (8) just by replacing a formula with one that is equivalent.

Actually, stepping back, the same argument can be made to go from (9) to (8) instead of from (8) to (9), because (10) is an equivalence. Both ways of using [∪] are perfectly fine. Although the direction that gets rid of the ∪ operator tends to be much more useful, because it made progress (getting rid of an HP operator).

Yet axiom [∪] can also be useful in many more situations. For example, axiom [∪] also justifies the inference

$$[\cup]\frac{[\alpha]A \wedge [\beta]A \to B}{[\alpha \cup \beta]A \to B}$$

which follows from the left-to-right implication of axiom [∪].

A general principle behind the dℒ axioms is most noticeable in axiom [∪]. All equivalence axioms of dℒ are primarily intended to be used by reducing the formula on the left to the (structurally simpler) formula on the right. Such a reduction symbolically decomposes a property of a more complicated system into separate properties of easier fragments α and β. While we might end up with more subproperties (like we do in the case of axiom [∪]), each of them is structurally simpler, because it involves less program operators. This decomposition of systems into their fragments makes the problem tractable and is good for scalability purposes, because it reduces the study of complex systems successively to a study of many but smaller subsystems of which there are only finitely many. For these symbolic structural decompositions, it is very helpful that dℒ is a full logic that is closed under all logical operators [Pla15a], including disjunction and conjunction, for then both sides in [∪] are dℒ formulas again (unlike in Hoare logic [Hoa69]). This also turns out to be an advantage for computing invariants [PC08, PC09, Pla10, GP14], which will be discussed much later in this course.

4 Soundness

The definition of soundness in Lemma 1 was not specific to axiom [∪], but applies to all dℒ axioms.

Definition 2 (Soundness)**.** An axiom is *sound* iff all its instances are valid.

From now on, every time we see a formula of the form $[\alpha \cup \beta]P$, we can remember that axiom [∪] knows a formula, namely $[\alpha]P \wedge [\beta]P$ that is equivalent to it. Whenever we find a formula of the form $[\gamma \cup \delta]Q$, we also remember that axiom [∪] knows a formula, namely $[\gamma]Q \wedge [\delta]Q$ that is equivalent to it, just by instantiation [Pla15b] of axiom [∪]. And the fact that axiom [∪] is sound ensures that we do not need to worry about whether such reasoning is correct every time we need it. Every instance of [∪] is sound.

[4] This will be made rigorous in a later lecture following contextual equivalence reasoning [Pla15b].

Once we know that [∪] is sound, we can treat it syntactically and mechanically and apply it as needed, like a machine would.

But because soundness is such a big deal (a *conditio sine qua non* in logic, i.e., something without which logic could not be), we will prove soundness of [∪] carefully, even if we almost already did in our informal argument above.

Proof of Lemma 1. The fact that axiom [∪] is sound can be proved as follows. Since $[\![\alpha \cup \beta]\!] = [\![\alpha]\!] \cup [\![\beta]\!]$, we have that $(\omega, \nu) \in [\![\alpha \cup \beta]\!]$ iff $(\omega, \nu) \in [\![\alpha]\!]$ or $(\omega, \nu) \in [\![\beta]\!]$. Thus, $\omega \in [\![[\alpha \cup \beta]P]\!]$ iff $\omega \in [\![[\alpha]P]\!]$ and $\omega \in [\![[\beta]P]\!]$. □

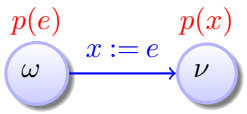
Why is soundness so critical? Well, because, without it, we could accidentally declare a system safe that is not in fact safe, which would defeat the whole purpose of verification and possibly put human lives in jeopardy when they are trusting their lives on an unsafe CPS. Unfortunately, soundness is actually not granted in all verification techniques for hybrid systems. But we will make it a point in this course to only ever use sound reasoning and scrutinizing all verification for soundness right away. Soundness is something that is comparably easy to establish in logic and proof approaches, because it localizes into the separate study of soundness of each of its axioms.

5 Dynamic Axioms for Assignments

Axiom [∪] allows us to understand and handle $[\alpha \cup \beta]$ properties. If we find similar axioms for all the other operators of hybrid programs, then we have a way of handling all other hybrid programs, too [Pla12b].

Consider discrete assignments. Recall from Lecture 4 on Safety & Contracts that:

$$[\![x := e]\!] = \{(\omega, \nu) \ : \ \nu = \omega \text{ except that } [\![x]\!]\nu = [\![e]\!]\omega\}$$



Lemma 3 ([:=] soundness)**.** *The assignment axiom is sound:*

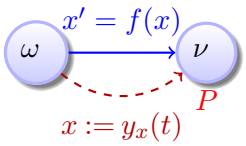
$$[:=] \quad [x := e]p(x) \leftrightarrow p(e)$$

The assignment axiom [:=] expresses that $p(x)$ is true after the discrete assignment assigning e to x iff $p(e)$ has already been true before that change, since the assignment $x := e$ will change around the value of variable x to the value of e [Pla15b].

6 Dynamic Axioms for Differential Equations

Recall from Lecture 3 on Choice & Control that

$$[\![x' = f(x)\,\&\,Q]\!] = \{(\varphi(0), \varphi(r)) \;:\; \varphi(t) \models x' = f(x) \text{ and } \varphi(t) \models Q \text{ for all } 0 \leq t \leq r \text{ for a solution } \varphi : [0, r] \to \mathcal{S} \text{ of any duration } r\}$$



One possible approach of proving properties of differential equations is to work with a solution if one is available (and expressible in the logic). After all, the first thing you learned about what to di with differential equations is probably to solve them.

Lemma 4 ([′] soundness)**.** *The solution axiom schema is sound:*

$$[']\quad [x' = f(x)]P \leftrightarrow \forall t{\geq}0\, [x := y(t)]P \quad (y'(t) = f(y))$$

where $y(\cdot)$ is the solution of the symbolic initial-value problem $y'(t) = f(y), y(0) = x$.

Solution $y(\cdot)$ is unique since $f(x)$ is smooth (Lecture 2). Given such a solution $y(\cdot)$, continuous evolution along differential equation $x' = f(x)$ can be replaced by a discrete assignment $x := y(t)$ with an additional quantifier for the evolution time t. It goes without saying that variables like t are fresh in [′] and other axioms and proof rules. Notice that conventional initial-value problems are numerical with concrete numbers $x \in \mathbb{R}^n$ as initial values, not symbols x [Wal98]. This would not be enough for our purpose, because we need to consider all states in which the system could start, which may be uncountably many. That is why axiom [′] solves one symbolic initial-value problem, instead, because we could hardly solve uncountable many numerical initial-value problems.

Note 9 (Discrete vs. continuous dynamics)**.** *Notice something rather intriguing and peculiar about axiom [′]. It relates a property of a continuous system to a property of a discrete system. The HP on the left-hand side describes a smoothly changing continuous process, while the right-hand side describes an abruptly, instantaneously changing discrete process. Still, their respective properties coincide, thanks to the time quantifier. This is the beginning of an astonishingly intimate relationship of discrete and continuous dynamics [Pla12b].*

What we have so far about the dynamics of differential equations does not yet help us prove properties of differential equations with evolution domain constraints (a.k.a. continuous programs) $x' = f(x)\,\&\,Q$. It also does not yet tell us what to do if we cannot solve the differential equation or if the solution is too complicated. We will get to

those matters in more detail in later lectures. Just briefly note that evolution domain constraints can be handled as well by adding a condition checking that the evolution domain was always true until the point in time of interest:

$$[']\ \ [x' = f(x)\,\&\,Q]P \leftrightarrow \forall t{\geq}0\,\big((\forall 0{\leq}s{\leq}t\,[x := y(s)]Q) \rightarrow [x := y(t)]P\big)$$

The effect of the additional constraint on Q is to restrict the continuous evolution such that its solution $y(s)$ remains in the evolution domain Q at all intermediate times $s \leq t$. This constraint simplifies to $true$ if the evolution domain Q is $true$, which makes sense, because there are no special constraints on the evolution (other than the differential equations) if the evolution domain is described by $true$, hence the full state space.

7 Dynamic Axioms for Tests

Recall from Lecture 3 on Choice & Control that

$$[\![?Q]\!] = \{(\omega,\omega)\ :\ \omega \in [\![Q]\!]\}$$

Lemma 5 ([?] soundness)**.** *The test axiom is sound:*

$$[?]\ \ [?Q]P \leftrightarrow (Q \rightarrow P)$$

Tests in $[?Q]P$ are proven by assuming that the test succeeds with an implication in axiom [?], because test $?Q$ can only make a transition when condition Q actually holds true. In states where test Q fails, no transition is possible and the failed attempt to run the system is discarded. If no transition exists for an HP α, there is nothing to show for $[\alpha]P$ formulas, because their semantics requires P to hold in all states reachable by running α, which is vacuously true if no states are reachable. From left to right, axiom [?] for dL formula $[?Q]P$ assumes that formula Q holds true (otherwise there is no transition and thus nothing to show) and shows that P holds after the resulting no-op. The converse implication from right to left is by case distinction. Either Q is false, then $?Q$ cannot make a transition and there is nothing to show. Or Q is true, but then also P is true according to the implication.

8 Dynamic Axioms for Sequential Compositions

For sequential compositions $\alpha;\beta$, Sect. 2 proposed the use of an intermediate condition E characterizing all intermediate states between α and β by way of the following proof

rule:

$$\text{G[;]}\ \frac{A \to [\alpha]E \quad E \to [\beta]B}{A \to [\alpha;\beta]B}$$

This proof rule can sometimes be useful, but it has one blatant annoyance compared to the simplicity and elegance of axiom [∪]. When using rule G[;] from the desired conclusion to the premises, it does not say how to choose the intermediate condition E. Using G[;] successfully requires us to find the right intermediate condition E, for if we don't, the proof won't succeed as we have seen in the Appendix of Lecture 4. That is a bit much if we have to invent a useful intermediate condition E for every single sequential composition in a CPS.

Fortunately, differential dynamic logic provides a much better way that we also identify by investigating the dynamical system resulting from $\alpha;\beta$ and its induced Kripke structure. Recall from Lecture 3 on Choice & Control that

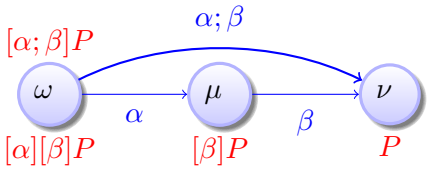
$$[\![\alpha;\beta]\!] = [\![\alpha]\!] \circ [\![\beta]\!] \overset{\text{def}}{=} \{(\omega,\nu) : (\omega,\mu) \in [\![\alpha]\!], (\mu,\nu) \in [\![\beta]\!]\} \tag{11}$$

By its semantics, the dL formula $[\alpha;\beta]P$ is true in a state ω iff P is true in all states that $\alpha;\beta$ can reach according to $[\![\alpha;\beta]\!]$ from ω, i.e. all those states for which $(\omega,\nu) \in [\![\alpha;\beta]\!]$. Which states are those? And how do they relate to the states reachable by α or by β alone? They do not relate to those in a way that is as direct as for axiom [∪]. But they still relate, and they do so by way of (11).

Postcondition P has to be true in all states reachable by $\alpha;\beta$ from ω for $[\alpha;\beta]P$ to be true at ω. By (11), those are exactly the states ν to which we can get by running β from an intermediate state μ to which we have gotten from ω by running α. Thus, for $[\alpha;\beta]P$ to be true at ω it is necessary that P holds in all states ν to which we can get by running β from an intermediate state μ to which we can get by running β from ω. Consequently, $[\alpha;\beta]P$ is only true at ω if $[\beta]P$ holds in all those intermediate states μ to which we can get from ω by running α. How do we characterize those states? And how can we then express these thoughts in a single logical formula of dL?

Before you read on, see if you can find the answer for yourself.

If we want to express that $[\beta]P$ holds in all states μ to which we can get to from ω by running α, then that is exactly what truth of dL formula $[\alpha][\beta]P$ at ω means, because this is the semantics of the modality $[\beta]$.



Consequently,

$$\omega \in [\![[\alpha][\beta]P \to [\alpha;\beta]P]\!]$$

Reexamining our argument backwards, we see that the converse implication also holds

$$\omega \in [\![[\alpha;\beta]P \to [\alpha][\beta]P]\!]$$

The same argument works for all ω, so both implications are even valid.

Lemma 6 ([;] soundness)**.** *The composition axiom is sound:*

$$[;]\ \ [\alpha;\beta]P \leftrightarrow [\alpha][\beta]P$$

Proof. Since $[\![\alpha;\beta]\!] = [\![\alpha]\!] \circ [\![\beta]\!]$, we have that $(\omega,\nu) \in [\![\alpha;\beta]\!]$ iff $(\omega,\mu) \in [\![\alpha]\!]$ and $(\mu,\nu) \in [\![\beta]\!]$ for some intermediate state μ. Hence, $\omega \in [\![[\alpha;\beta]P]\!]$ iff $\mu \in [\![[\beta]P]\!]$ for all μ with $(\omega,\mu) \in [\![\alpha]\!]$. That is $\omega \in [\![[\alpha;\beta]P]\!]$ iff $\omega \in [\![[\alpha][\beta]P]\!]$. □

Sequential compositions are proven using nested modalities in axiom [;]. From right to left: If, after all α-runs, it is the case that all β-runs lead to states satisfying P (i.e., $[\alpha][\beta]P$ holds), then all runs of the sequential composition $\alpha;\beta$ lead to states satisfying P (i.e., $[\alpha;\beta]P$ holds), because $\alpha;\beta$ cannot go anywhere but following α through some intermediate state to running β. The converse implication uses the fact that if after all α-runs all β-runs lead to P (i.e., $[\alpha][\beta]P$), then all runs of $\alpha;\beta$ lead to P (that is, $[\alpha;\beta]P$), because the runs of $\alpha;\beta$ are exactly those that first do any α-run, followed by any β-run. Again, it is crucial that dL is a full logic that considers reachability statements as modal operators, which can be nested, for then both sides in axiom [;] are dL formulas.

Axiom [;] directly explains sequential composition $\alpha;\beta$ in terms of a structurally simpler formula, one with nested modal operators but simpler hybrid programs. Again, using axiom [;] by reducing occurrences of its left-hand side to its right-hand side decomposes the formula into structurally simpler pieces, thereby making progress. One of the many ways of using axiom [;] is, therefore, captured in the following proof rule:

$$\text{R8}\ \frac{A \to [\alpha][\beta]B}{A \to [\alpha;\beta]B}$$

Comparing rule R8 to rule G[;], the new rule R8 is much easier to apply, because it does not require us to first identify and provide an intermediate condition E like rule G[;]

would. It also does not branch into two premises, which helps keeping the proof lean. Is there a way of reuniting R8 with G[;] by using the expressive power of dℒ?

Before you read on, see if you can find the answer for yourself.

Yes, indeed, there is a very smart choice for the intermediate condition E that makes G[;] behave almost as the more efficient R8 would. The clever choice $E \overset{\text{def}}{\equiv} [\beta]B$:

$$\frac{A \to [\alpha][\beta]B \quad [\beta]B \to [\beta]B}{A \to [\alpha;\beta]B}$$

which trivializes the right premise, because all formulas imply themselves, and makes the left premise identical to that of R8. Consequently, differential dynamic logic internalizes ways of expressing necessary and possible properties of hybrid programs and makes both first-class citizens in the logic. That cuts down on the amount of input that is needed when conducting proofs.

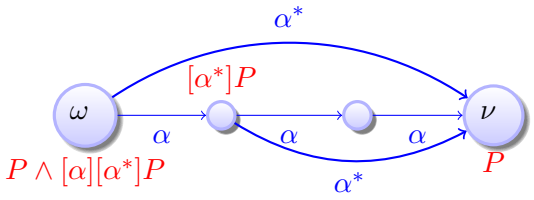
9 Unwinding Axioms for Loops

Recall from Lecture 3 on Choice & Control that

$$[\![\alpha^*]\!] = \bigcup_{n\in\mathbb{N}} [\![\alpha^n]\!] \qquad \text{with} \quad \alpha^{n+1} \equiv \alpha^n;\alpha \text{ and } \alpha^0 \equiv ?true$$

How could we prove properties of loops such as $[\alpha^*]P$? Is there a way of reducing properties of loops to properties of simpler systems in similar ways as the other axioms of differential dynamic logic?

Before you read on, see if you can find the answer for yourself.



Lemma 7 ([*] soundness). *The iteration axiom is sound:*

$$[^*]\ \ [\alpha^*]P \leftrightarrow P \wedge [\alpha][\alpha^*]P$$

Axiom [*] is the iteration axiom, which partially unwinds loops. It uses the fact that P always holds after repeating α (i.e., $[\alpha^*]P$), if P holds at the beginning (for P holds after zero repetitions then), and if, after one run of α, P holds after every number of repetitions of α, including zero repetitions (i.e., $[\alpha][\alpha^*]P$). So axiom [*] expresses that $[\alpha^*]P$ holds iff P holds immediately and after one or more repetitions of α.

The same axiom [*] can be used to unwind loops $N \in \mathbb{N}$ times, which corresponds to Bounded Model Checking [CBRZ01]. If the formula is not valid, a bug has been found, otherwise N increases. An obvious issue with this simple approach is that we can never stop increasing N if the formula is actually valid, because we can never find a bug then. A later lecture will discuss proof techniques for repetitions based on loop invariants that are not subject to this issue. In particular, axiom [*] is characteristically different from the other axioms discussed in this lecture. Unlike the other axioms, [*] does not exactly get rid of the formula on the left-hand side. It just puts it in a different syntactic place, which does not sound like much progress.[5]

10 A Proof of a Short Bouncing Ball

Now that we have understood so many axioms and proof rules, let us use them to prove the (single-hop) bouncing ball (2):

$$0 \leq x \wedge x = H \wedge v = 0 \wedge g > 0 \wedge 1 \geq c \geq 0 \rightarrow \\ \left[x' = v, v' = -g \,\&\, h \geq 0; (?x = 0; v := -cv \cup ?x \neq 0)\right](0 \leq x \wedge x \leq H) \quad (2)$$

Before proceeding, let's modify the hybrid program ever so subtly in two ways so that there's no more evolution domains, just so that we do not yet have to deal with the evolution domains yet. We boldly drop the evolution domain constraint and make up for it by modifying the condition in the second test:

[5] With a much more subtle and tricky analysis, it is possible to prove that [*] still makes sufficient progress [Pla14]. But this is out of scope for our course.

$$0 \leq x \wedge x = H \wedge v = 0 \wedge g > 0 \wedge 1 \geq c \geq 0 \rightarrow \\ \big[x' = v, v' = -g; (?x = 0; v := -cv \cup ?x \geq 0)\big]\,(0 \leq x \wedge x \leq H) \quad (12)$$

Hold on, why is that okay? Doesn't our previous investigation say that Quantum could suddenly fall through the cracks in the floor if physics insists on evolving for hours before giving the poor bouncing ball controller a chance to react? To make sure Quantum does not panic in light of this threat, solve Exercise 8 to investigate.

To fit things on the page easily, abbreviate

$$A \overset{\text{def}}{\equiv} 0 \leq x \wedge x = H \wedge v = 0 \wedge g > 0 \wedge 1 \geq c \geq 0$$
$$B(x,v) \overset{\text{def}}{\equiv} 0 \leq x \wedge x \leq H$$
$$(x'' = -g) \overset{\text{def}}{\equiv} (x' = v, v' = -g)$$

With these abbreviations, (12) turns into

$$A \rightarrow [x'' = -g; (?x = 0; v := -cv \cup ?x \geq 0)]B(x,v)$$

Let there be proof for bouncing balls:

$$\begin{array}{ll}
 & A \rightarrow \forall t{\geq}0\,\big((H - \frac{g}{2}t^2 = 0 \rightarrow B(H - \frac{g}{2}t^2, -c(-gt))) \wedge (H - \frac{g}{2}t^2 \geq 0 \rightarrow B(H - \frac{g}{2}t^2, -gt))\big) \\
[:=] & A \rightarrow \forall t{\geq}0\,[x := H - \frac{g}{2}t^2]\big((x = 0 \rightarrow B(x, -c(-gt))) \wedge (x \geq 0 \rightarrow B(x, -gt))\big) \\
[:=] & A \rightarrow \forall t{\geq}0\,[x := H - \frac{g}{2}t^2][v := -gt]\big((x = 0 \rightarrow B(x, -cv)) \wedge (x \geq 0 \rightarrow B(x, v))\big) \\
[;] & A \rightarrow \forall t{\geq}0\,[x := H - \frac{g}{2}t^2; v := -gt]\big((x = 0 \rightarrow B(x, -cv)) \wedge (x \geq 0 \rightarrow B(x, v))\big) \\
['] & A \rightarrow [x'' = -g]\big((x = 0 \rightarrow B(x, -cv)) \wedge (x \geq 0 \rightarrow B(x, v))\big) \\
[:=] & A \rightarrow [x'' = -g]\big((x = 0 \rightarrow [v := -cv]B(x, v)) \wedge (x \geq 0 \rightarrow B(x, v))\big) \\
[?],[?] & A \rightarrow [x'' = -g]\big([?x = 0][v := -cv]B(x, v) \wedge [?x \geq 0]B(x, v)\big) \\
[;] & A \rightarrow [x'' = -g]\big([?x = 0; v := -cv]B(x, v) \wedge [?x \geq 0]B(x, v)\big) \\
[\cup] & A \rightarrow [x'' = -g][?x = 0; v := -cv \cup ?x \geq 0]B(x, v) \\
[;] & A \rightarrow [x'' = -g; (?x = 0; v := -cv \cup ?x \geq 0)]B(x, v)
\end{array}$$

The dℒ axioms indicated on the left justify that the dℒ formulas in the two adjacent rows are equivalent. Since each step in this proof is justified by using a dℒ axiom, the conclusion at the very bottom of this derivation is proved if the premise at the very top can be proved, because truth then inherits from the top to the bottom. That premise

$$A \rightarrow \forall t{\geq}0\,\big((H - \frac{g}{2}t^2 = 0 \rightarrow B(H - \frac{g}{2}t^2, -c(-gt))) \wedge (H - \frac{g}{2}t^2 \geq 0 \rightarrow B(H - \frac{g}{2}t^2, -gt))\big)$$

expands out to a formula of first-order real arithmetic by expanding the abbreviations:

$$0 \leq x \wedge x = H \wedge v = 0 \wedge g > 0 \wedge 1 \geq c \geq 0 \rightarrow \\ \forall t{\geq}0\,\big((H - \frac{g}{2}t^2 = 0 \rightarrow 0 \leq H - \frac{g}{2}t^2 \wedge H - \frac{g}{2}t^2 \leq H) \\ \wedge (H - \frac{g}{2}t^2 \geq 0 \rightarrow 0 \leq H - \frac{g}{2}t^2 \wedge H - \frac{g}{2}t^2 \leq H)\big)$$

In this case, this remaining premise can be easily seen to be valid. The assumption $H - \frac{g}{2}t^2 = 0 \rightarrow \ldots$ in the middle line directly implies the first conjunct that appears in its respective right-hand side

$$0 \leq H - \frac{g}{2}t^2 \wedge H - \frac{g}{2}t^2 \leq H$$

and reduces the remaining second conjunct to $0 \leq H$, which the assumption in the first line assumed ($0 \leq x = H$). Similarly, the assumption $H - \frac{g}{2}t^2 \geq 0$ of the last line implies the first conjunct of its right-hand side

$$0 \leq H - \frac{g}{2}t^2 \wedge H - \frac{g}{2}t^2 \leq H$$

and its second conjunct holds by assumption $g > 0$ from the first line and the real arithmetic fact that $t^2 \geq 0$.

How exactly first-order logic and first-order real arithmetic formulas such as this one can be proved in general, however, is an interesting topic for a later lecture. For now, we are happy to report that we have just formally verified our very first CPS. We have found a proof of (12). Exciting!

Okay, admittedly, the CPS we just verified was only a bouncing ball. And all we know about it now is that it won't fall through the cracks in the ground nor jump high up to the moon. But most big steps for mankind start with a small step by someone.

Yet, before we get too carried away in the excitement, we still need to remember that (12) is just a single-hop bouncing ball. So there's still an argument to be made about what happens if the bouncing ball repeats. And a rather crucial argument too, because bouncing balls let loose in the air tend not to jump any higher anyhow without hitting the ground first, which is where the model (12) stops prematurely, because it is missing a repetition. So let's put worrying about loops on the agenda for an upcoming lecture.

Yet, there's one more issue with the proof for the bouncing ball that we derived. It works in a somewhat undisciplined chaotic way, by using $\mathsf{d}\mathcal{L}$ axioms all over the place. This liberal proof style can be useful for manual proofs and creative shortcuts. Albeit, since the $\mathsf{d}\mathcal{L}$ axioms are sound, even such a liberal proof is a still proof. And liberal proofs could even be very creative. But liberal proofs are also somewhat unfocused and non-systematic, which makes them unreasonable for automation purposes and also tends to get people lost if the problems at hand are more complex than the single-hop bouncing ball. That is the reason why we will investigate more focused, more systematic, and more algorithmic proofs in the next lecture.

The other thing to observe is that the above proof, however liberal it might have been, already had more structure to it than we made explicit. This structure will be uncovered in the next lecture.

11 Summary

The differential dynamic logic axioms that we have seen in this lecture are summarized in Note 13. There are further axioms and proof rules of differential dynamic logic that

later lectures will examine [Pla12d, Pla12b, Pla15b], but the reasoning principles and axioms identified here are fundamental and we will carry them with us throughout the whole course.

Note 13 (Summary of differential dynamic logic axioms from this lecture)**.** *The following axioms of* dL *are sound, i.e., valid and so are all their instances:*

[:=] $[x := e]p(x) \leftrightarrow p(e)$

[?] $[?Q]P \leftrightarrow (Q \rightarrow P)$

[′] $[x' = f(x)]P \leftrightarrow \forall t{\geq}0\, [x := y(t)]P$ $\qquad (y'(t) = f(y))$

[∪] $[\alpha \cup \beta]P \leftrightarrow [\alpha]P \wedge [\beta]P$

[;] $[\alpha;\beta]P \leftrightarrow [\alpha][\beta]P$

[*] $[\alpha^*]P \leftrightarrow P \wedge [\alpha][\alpha^*]P$

Exercises

Exercise 1. Explain why the subtle transformation from (2) to (12) was okay *in this case*.

Exercise 2. Identify which of the assumptions of (12) are actually required for the proof of (12). Which formulas could we have dropped from $0 \leq x \wedge x = H \wedge v = 0 \wedge g > 0 \wedge 1 \geq c \geq 0$ and still be able to prove

$$0 \leq x \wedge x = H \wedge v = 0 \wedge g > 0 \wedge 1 \geq c \geq 0 \rightarrow [x'' = -g; (?x = 0; v := -cv \cup ?x \geq 0)](0 \leq x \wedge x \leq H)$$

Exercise 3. Develop possible axioms for differential equations with evolution domains similar to [′]. That is, develop an axiom for $[x' = f(x)\,\&\,Q]P$. As in [′], you can assume to have a unique solution for the corresponding symbolic initial-value problem.

Exercise 4. Would the following be a sound axiom? Proof or disprove.

$$[x' = \theta\,\&\,Q]P \leftrightarrow \forall t{\geq}0\, \forall 0{\leq}s{\leq}t\, \big([x := y(s)]Q \rightarrow [x := y(t)]P\big)$$

Exercise 5. All axioms need to be proved to be sound. These lecture notes only did a proper proof for [∪] and [;]. Turn the informal arguments for the other axioms into proper soundness proofs using the semantics of dL formulas.

Exercise 6. Would the following be a useful replacement for the [*] axiom?

$$[\alpha^*]P \leftrightarrow P \wedge [\alpha^*]P$$

Exercise 7. This lecture identified axioms for all formulas of the form $[\alpha]P$ but none for formulas of the form $\langle\alpha\rangle P$. Identify and justify these missing axioms. Explain how they relate to the ones given in Note 13.

Exercise 8 (Give bouncing ball back its evolution domain). Explain why the transformation from (1) to (12) was okay in this case.

References

[CBRZ01] Edmund M. Clarke, Armin Biere, Richard Raimi, and Yunshan Zhu. Bounded model checking using satisfiability solving. *Form. Methods Syst. Des.*, 19(1):7–34, 2001.

[GP14] Khalil Ghorbal and André Platzer. Characterizing algebraic invariants by differential radical invariants. In Erika Ábrahám and Klaus Havelund, editors, *TACAS*, volume 8413 of *LNCS*, pages 279–294. Springer, 2014. doi:10.1007/978-3-642-54862-8_19.

[Hoa69] Charles Antony Richard Hoare. An axiomatic basis for computer programming. *Commun. ACM*, 12(10):576–580, 1969.

[LIC12] *Proceedings of the 27th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2012, Dubrovnik, Croatia, June 25–28, 2012*. IEEE, 2012.

[MP16] Stefan Mitsch and André Platzer. ModelPlex: Verified runtime validation of verified cyber-physical system models. *Form. Methods Syst. Des.*, 2016. Special issue of selected papers from RV'14. doi:10.1007/s10703-016-0241-z.

[PC08] André Platzer and Edmund M. Clarke. Computing differential invariants of hybrid systems as fixedpoints. In Aarti Gupta and Sharad Malik, editors, *CAV*, volume 5123 of *LNCS*, pages 176–189. Springer, 2008. doi:10.1007/978-3-540-70545-1_17.

[PC09] André Platzer and Edmund M. Clarke. Computing differential invariants of hybrid systems as fixedpoints. *Form. Methods Syst. Des.*, 35(1):98–120, 2009. Special issue for selected papers from CAV'08. doi:10.1007/s10703-009-0079-8.

[Pla08] André Platzer. Differential dynamic logic for hybrid systems. *J. Autom. Reas.*, 41(2):143–189, 2008. doi:10.1007/s10817-008-9103-8.

[Pla10] André Platzer. *Logical Analysis of Hybrid Systems: Proving Theorems for Complex Dynamics*. Springer, Heidelberg, 2010. doi:10.1007/978-3-642-14509-4.

[Pla12a] André Platzer. A complete axiomatization of quantified differential dynamic logic for distributed hybrid systems. *Log. Meth. Comput. Sci.*, 8(4):1–44, 2012. Special issue for selected papers from CSL'10. doi:10.2168/LMCS-8(4:17)2012.

[Pla12b] André Platzer. The complete proof theory of hybrid systems. In LICS [LIC12], pages 541–550. doi:10.1109/LICS.2012.64.

[Pla12c] André Platzer. Dynamic logics of dynamical systems. *CoRR*, abs/1205.4788, 2012. arXiv:1205.4788.

[Pla12d] André Platzer. Logics of dynamical systems. In LICS [LIC12], pages 13–24. doi:10.1109/LICS.2012.13.

[Pla14] André Platzer. Differential game logic. *CoRR*, abs/1408.1980, 2014. arXiv:1408.1980.

[Pla15a] André Platzer. Differential game logic. *ACM Trans. Comput. Log.*, 17(1):1:1–1:51, 2015. doi:10.1145/2817824.

[Pla15b] André Platzer. A uniform substitution calculus for differential dynamic logic. In Amy Felty and Aart Middeldorp, editors, *CADE*, volume 9195 of *LNCS*, pages 467–481. Springer, 2015. doi:10.1007/978-3-319-21401-6_32.

[SKSC98] Danbing Seto, Bruce Krogh, Lui Sha, and Alongkrit Chutinan. The Simplex architecture for safe online control system upgrades. In *ACC*, volume 6, pages 3504–3508, 1998.

[Wal98] Wolfgang Walter. *Ordinary Differential Equations*. Springer, 1998.