15-424: Foundations of Cyber-Physical Systems

Lecture Notes on Control Loops & Invariants

André Platzer

Carnegie Mellon University
Lecture 7

1 Introduction

Lecture 3 on Choice & Control demonstrated how important control is in CPS and that control loops are a very important feature for making this control happen. Without loops, CPS controllers are limited to short finite sequences of control actions, which are rarely sufficient to get our CPS anywhere. With loops, CPS controllers shine, because they can inspect the current state of the system, take action to control the system, let the physics evolve, and then repeat these steps in a loop over and over again to slowly get the state where the controller wants the system to be. Loops truly make feedback happen, by enabling a CPS to sense state and act in response to that over and over again. Think of programming a robot to drive on a highway. Would you be able to do that without some means of repetition or iteration as in repeated control? Probably not, because you'll need to write a CPS program that monitors the traffic situation frequently and reacts in response to what the other cars do on the highway. There's no way of telling ahead of time, how often the robot will need to change its mind when its driving a car on a highway.

Hybrid programs' way of exercising repetitive control actions is the repetition operator * that can be applied to any hybrid program α. The resulting hybrid program α^* repeats α any number of times, nondeterministically. That may be zero times or 1 time or 10 times or ....

Now, the flip side of the fact that control loops are responsible for a lot of the power of CPS is that they can also be tricky to analyze and fully understand. After all, what a system does in just one step is easier to get a handle on than to understand what it will do in the long run when the CPS is running for any arbitrary amount of time. This is the CPS analogue of the fact that ultra-short-term predictions are often much easier

than long-term predictions. It is easy to predict the weather a second into the future but much harder to predict next week's weather.

The main insight behind the analysis of loops in CPS is to reduce the (complicated) analysis of their long-term global behavior to a simpler analysis of their local behavior for one control cycle. This principle significantly reduces the analytic complexity of loops in CPS. It leverages invariants, i.e. aspects of the system behavior that do not change as time progresses, so that our analysis can rely on them no matter how long the system already evolved. Invariants turn out to also lead to an important design principle for CPS, even more so than in programs [PCL11]. The significance of invariants in understanding CPS is not a coincidence, because the study of invariants (just of other mathematical structures) is also central to a large body of mathematics.

More information can be found in [Pla12b, Pla12a] as well as [Pla10, Chapter 2.5.2,2.5.4].

The most important learning goals of this lecture are:

Modeling and Control: We develop a deeper understanding of control loops as a core principle behind CPS that is ultimately underlying all feedback mechanisms in CPS control. This lecture also intensifyies our understanding of the dynamical aspects of CPS and how discrete and continuous dynamics interact.

Computational Thinking: This lecture extends the rigorous reasoning approach from Lecture 5 on Dynamical Systems & Dynamic Axioms to systems with repetitions. This lecture is devoted to the development of rigorous reasoning techniques for CPS models with repetitive control loops or other loopy behavior, a substantially nontrivial problem in theory and practice. Without understanding loops, there is no hope of understanding the repetitive behavior of feedback control principles that are common to almost all CPS. Understanding such behavior can be tricky, because so many things can change in the system and its environment over the course of the runtime of even just a few lines of code if that program runs repeatedly to control the behavior of a CPS. That is why the study of *invariants*, i.e. properties that do not change throughout the execution of the system are crucial for their analysis. Invariants constitute the single most insightful and most important piece of information about a CPS. As soon as we understand the invariants of a CPS, we almost understand everything about it and will even be in a position to design the rest of the CPS around this invariant, a process known as design-by-invariant principle. Identifying and expressing invariants of CPS models will be a part of this lecture as well.

The first part of the lecture shows a careful and systematic development of the invariants, discussing some proof rules and proof principles of more general interest along the way. The second part of the lecture focuses on invariants.

Another aspect of today's lecture is the important concept of global proof rules, which have global premises rather than the local premises from the previous sequent proof rules.

CPS Skills: We will develop a better understanding of the semantics of CPS models by understanding the core aspects of repetition and relating its semantics to cor-

responding reasoning principles. This understanding will lead us to develop a higher level of intuition for the operational effects involved in CPS by truly understanding what control loops fundamentally amount to.

2 Control Loops

Recall the little acrophobic bouncing ball from Lecture 4 on Safety & Contracts.

$$\begin{array}{l}\texttt{@requires}(0 \leq x \wedge x = H \wedge v = 0)\\ \texttt{@requires}(g > 0 \wedge 1 \geq c \geq 0)\\ \texttt{@ensures}(0 \leq x \wedge x \leq H)\\ \big(x' = v, v' = -g \,\&\, x \geq 0;\\ \quad \texttt{if}(x = 0)\, v := -cv\big)^*\end{array} \tag{1}$$

The contracts above have been augmented with the ones that we have identified in Lecture 4 by converting the initial contract specification into a logical formula in differential dynamic logic and then identifying the required assumptions to make it true in all states:

$$0 \leq x \wedge x = H \wedge v = 0 \wedge g > 0 \wedge 1 \geq c \geq 0 \rightarrow \\ \big[\big(x' = v, v' = -g \,\&\, x \geq 0;\ \texttt{if}(x = 0)\, v := -cv\big)^*\big]\,(0 \leq x \wedge x \leq H) \tag{2}$$

Because we did not want to be bothered by the presence of the additional `if-then-else` operator, which is not officially part of the minimal set of operators that differential dynamic logic dℒ provides, we simplified (2) to:

$$0 \leq x \wedge x = H \wedge v = 0 \wedge g > 0 \wedge 1 \geq c \geq 0 \rightarrow \\ \big[\big(x' = v, v' = -g \,\&\, x \geq 0;\ (?x = 0; v := -cv \cup ?x \neq 0)\big)^*\big]\,(0 \leq x \wedge x \leq H) \tag{3}$$

In Lecture 4, we had an informal understanding why (3) is valid (true in all states), but no formal proof, albeit we proved a much simplified version of (3) in which we simply threw away the loop. Such ignorance is clearly not a correct way of understanding loops. Let's make up for that now by properly proving (3) in the dℒ calculus.

Yet, before going for a proof of this bouncing ball property, however much the bouncing ball may long for it, let us first take a step back and understand the role of loops in more general terms. Their semantics has been explored in Lecture 3 on Choice & Control and more formally in Lecture 5 on Dynamical Systems & Dynamic Axioms.

The little bouncing ball had a loop in which physics and its bounce control alternated. The bouncing ball desperately needs a loop for it wouldn't know ahead of time how often it would bounce today. When falling from great heights, it bounces quite a bit. The bouncing ball also has a controller, albeit a rather impoverished one. All it could do is inspect the current height, compare it to the ground floor (at height 0) and, if $x = 0$, flip its velocity vector around after a little damping by factor c. That is not a whole lot of

flexibility for control choices, but the bouncing ball was still rather proud to serve such an important role in controlling the ball's behavior. Indeed, without the control action, the ball would never bounce back from the ground but would keep on falling forever—what a frightful thought for the acrophobic bouncing ball. On second thought, the ball would, actually, not even fall for very long without its controller, because of the evolution domain $x \geq 0$ for physics $x'' = -g \,\&\, x \geq 0$, which would only allow physics to evolve for time zero if the ball is already at height 0, because gravity would otherwise try to pull it further down, except that the $x \geq 0$ constraint won't have it. So, in summary, without the bouncing ball's control statement, it would simply fall and then lie flat on the ground without time being allowed to proceed. That would not sound very reassuring and certainly not as much fun as bouncing back up, so the bouncing ball is really quite proud of its control.

This principle is not specific to the bouncing ball, but, rather, quite common in CPS. The controller performs a crucial task, without which physics would not evolve in the way that we want it to. After all, if physics did already always do what we want it to without any input from our side, we would not need a controller for it in the first place. Hence, control is crucial and understanding and analyzing its effect on physics one of the primary responsibilities in CPS.

Before proving (3), we apply one more simplification that we have also done in Lecture 5, just to save space on the page. We boldly drop the evolution domain constraint and make up for it by modifying the condition in the second test (Exercise 1):

$$0 \leq x \wedge x = H \wedge v = 0 \wedge g > 0 \wedge 1 \geq c \geq 0 \rightarrow \\ \left[\left(x' = v, v' = -g;\ (?x = 0; v := -cv \cup ?x \geq 0)\right)^*\right](0 \leq x \wedge x \leq H) \quad (4)$$

Hold on, why is that okay? Doesn't our previous investigation say that the ball 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 the bouncing ball does not panic in light of this threat, solve Exercise 1 to investigate.

But, even if we took the evolution domain constraint away from it, notice what the bouncing ball in (4) proudly possess in comparison to the impoverished single-hop ball from Lecture 5: it still has a repetition. And the little acrophobic bouncing ball is certainly mighty proud of its repetition, for this is the only way it could ever bounce and bounce again rather than just bounce and have the world end right after.

3 Proofs of Loops

There is a loop in the HP of the modality in (4). As we have seen, its behavior is crucial to the bouncing ball. So let's prove to understand what it does and to see whether we have to be just as nervous as the bouncing ball about losing it to the earth (if postcondition $0 \leq x$ is not ensured) or to the sky (if $x \leq H$ is not ensured).

Abbreviations have served us well in trying to keep proofs onto one page:

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

With these abbreviations, the bouncing ball formula (4) turns into:

$$A_{x,v} \rightarrow [(x'' = -g; (?x = 0; v := -cv \cup ?x \geq 0))^*]B_{x,v} \tag{4}$$

This formula is swiftly turned into the sequent at the top using proof rule →r:

$$\rightarrow\text{r}\ \frac{A_{x,v} \vdash [(x'' = -g; (?x = 0; v := -cv \cup ?x \geq 0))^*]B_{x,v}}{\vdash A_{x,v} \rightarrow [(x'' = -g; (?x = 0; v := -cv \cup ?x \geq 0))^*]B_{x,v}}$$

This leaves a loop to be worried about. Inspecting our dℒ proof rules from Lecture 6 on Truth & Proof there is exactly one that addresses loops:

$$([^{*n}])\ \frac{\phi \land [\alpha][\alpha^*]\phi}{[\alpha^*]\phi}$$

Using this one to continue the sequent derivation proceeds as follows:

$$[^{*n}]\text{r}\ \frac{\land\text{r}\ \dfrac{*\ \dfrac{}{A_{x,v} \vdash B_{x,v}} \quad [;]\text{r}\ \dfrac{A_{x,v} \vdash [x'' = -g][?x = 0; v := -cv \cup ?x \geq 0][(x'' = -g; (?x = 0; v := -cv \cup ?x \geq 0))^*]B_{x,v}}{A_{x,v} \vdash [x'' = -g; (?x = 0; v := -cv \cup ?x \geq 0)][(x'' = -g; (?x = 0; v := -cv \cup ?x \geq 0))^*]B_{x,v}}}{A_{x,v} \vdash B_{x,v} \land [x'' = -g; (?x = 0; v := -cv \cup ?x \geq 0)][(x'' = -g; (?x = 0; v := -cv \cup ?x \geq 0))^*]B_{x,v}}}{A_{x,v} \vdash [(x'' = -g; (?x = 0; v := -cv \cup ?x \geq 0))^*]B_{x,v}}$$

The left subgoal that results from using ∧r closes by very simple arithmetic. The right subgoal is more of a challenge to prove. We can solve the differential equation and proceed using [′]r, which will produce a quantifier that ∀r can handle and leaves us with a sequent that we need to consider further to prove.

4 Loops of Proofs

After a lot of proof effort, the above sequent prove continues so that the first modalities

$$\ldots [x'' = -g][?x = 0; v := -cv \cup ?x \geq 0]\psi$$

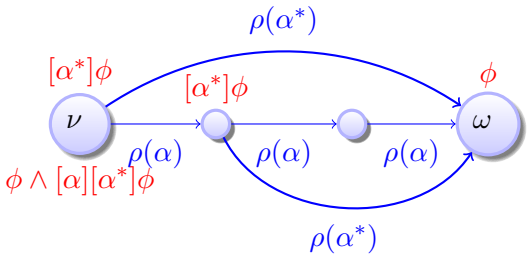
can be handled. But there is still a loop in the postcondition ψ, which is

$$\psi \equiv [(x'' = -g; (?x = 0; v := -cv \cup ?x \geq 0))^*]B_{x,v}$$

How can we prove that postcondition, then? Investigating our proof rules, there is exactly one that addresses loops: $[^{*n}]$r again. If we use $[^{*n}]$r again, what will happen?

Recall the loop semantics from Lecture 3 on Choice & Control and its unwinding axiom from Lecture 5 on Dynamical Systems & Dynamic Axioms:

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



Lemma 1 ([*] soundness). *The iteration axiom is sound:*

$$([^*])\ [\alpha^*]\phi \leftrightarrow \phi \wedge [\alpha][\alpha^*]\phi$$

Using proof rule $[^{*n}]$r on the succedent of a sequent has the same effect as using axiom [*] from left-hand side to right-hand side. Axiom [*] can be used to turn a formula

$$A \rightarrow [\alpha^*]B \tag{5}$$

into

$$A \rightarrow B \wedge [\alpha][\alpha^*]B$$

as we did in the bouncing ball half-proof above. What happens if we use that axiom [*] again?

Recall that, unlike sequent proof rules such as [*n]r, axioms do not say where they can be used, so we might as well use them anywhere in the middle of the formula. Hence using axiom [*] on the inner loop yields:

$$A \to B \wedge [\alpha](B \wedge [\alpha][\alpha^*]B)$$

Let's do that again and use the [*] axiom on the only occurrence of $[\alpha^*]B$ to obtain

$$A \to B \wedge [\alpha](B \wedge [\alpha](B \wedge [\alpha][\alpha^*]B)) \tag{6}$$

This is all very interesting but won't exactly get us any closer to a proof, because we could keep expanding the * star forever that way. How do we ever break out of this loop of never-ending proofs?

Before we get too disillusioned about our progress with [*] so far, notice that (6) still allows us to learn something about α and whether it always satisfies B when repeating α. Since [*] is an equivalence axiom, formula (6) still expresses the same thing as (5), i.e. that B always holds after repeating α when A was true in the beginning. Yet, (6) explicitly singles out the first 3 runs of α. Let's make this more apparent by recalling

Lemma 2 (Boxes distribute over conjunctions)**.** *The following is a sound axiom*

$$([]\wedge)\ [\alpha](B \wedge \psi) \leftrightarrow [\alpha]B \wedge [\alpha]\psi$$

Using this valid equivalence turns (6) into

$$A \to B \wedge [\alpha]B \wedge [\alpha][\alpha](B \wedge [\alpha][\alpha^*]B)$$

Using []∧ again gives us

$$A \to B \wedge [\alpha]B \wedge [\alpha]([\alpha]B \wedge [\alpha][\alpha][\alpha^*]B)$$

Using []∧ once more gives

$$A \to B \wedge [\alpha]B \wedge [\alpha][\alpha]B \wedge [\alpha][\alpha][\alpha][\alpha^*]B \tag{7}$$

Looking at it this way, (7) could be more useful than the original (5), because, even though both are equivalent, (7) explicitly singles out the fact that B has to hold initially, after doing α once, after doing α twice, and that $[\alpha^*]B$ has to hold after doing α three times. Even if we are not quite sure what to make of the latter $[\alpha][\alpha][\alpha][\alpha^*]B$, because it still involves a loop, we are quite certain how to understand and handle the first three:

$$A \to B \wedge [\alpha]B \wedge [\alpha][\alpha]B \tag{8}$$

If this formula is not valid, then, certainly, neither is its equivalent (7) and, thus, neither is the original (5). Hence, if we find a counterexample to (8), we disproved (7) and (5). That can actually be rather useful.

Yet, if (8) is still valid, we do not know whether (7) and (5) are, since they involve stronger requirements (B holds after any number of repetitions of α). What can we do then? Simply unroll the loop once more by using [*] on (6) to obtain

$$A \to B \wedge [\alpha](B \wedge [\alpha](B \wedge [\alpha](B \wedge [\alpha][\alpha^*]B))) \tag{9}$$

Or, equivalently, use axiom [*] on (7) to obtain the equivalent

$$A \to B \wedge [\alpha]B \wedge [\alpha][\alpha]B \wedge [\alpha][\alpha][\alpha](B \wedge [\alpha][\alpha^*]B) \tag{10}$$

By sufficiently many uses of axiom []∧, (9) and (10) are both equivalent to

$$A \to B \wedge [\alpha]B \wedge [\alpha][\alpha]B \wedge [\alpha][\alpha][\alpha]B \wedge [\alpha][\alpha][\alpha^*]B \tag{11}$$

which we can again examine to see if we can find a counterexample to the first part

$$A \to B \wedge [\alpha]B \wedge [\alpha][\alpha]B \wedge [\alpha][\alpha][\alpha]B$$

If yes, we disproved (5), otherwise we use [*] once more.

Note 3 (Bounded model checking)**.** *This process of iteratively unrolling a loop with either axiom [*] or rule [$*^n$]r and then checking the resulting (loop-free) conjuncts is called* Bounded Model Checking *and has been used very successfully, e.g., in the context of finite-state systems [CBRZ01]. The same principle can be useful to disprove properties of loops in differential dynamic logic by unwinding the loop, checking to see if the resulting formulas have counterexamples and, if not, unroll the loop once more.*

Suppose such a bounded model checking process has been followed to unroll the loop $N \in \mathbb{N}$ times. What can you conclude about the safety of the system?

If a counterexample is found or the formula can be disproved, then we are certain that the CPS is unsafe. If, instead, all but the last conjunct in the Nth unrolling of the loop are provable then the system will be safe for $N-1$ steps, but we cannot conclude anything about the safety of the system after $N-1$ steps.

5 Breaking Loops for Proofs

Proving properties of loops by unwinding them forever with [$*^n$]r is not a promising strategy, unless we find that the conjecture is not valid after a number of unwindings. Or unless we do not mind being busy with the proof forever for infinitely many proof steps (which would never get our acrophobic bouncing ball off the ground either with the confidence that a safety argument provides). One way or another, we will have to find a way to break the loop apart to complete our reasoning.

Consider the formula (11) again that we got from (5) by unwinding the loop with axiom [*] a number of times and then flattening the formula with the help of axiom []∧:

$$A \to B \wedge [\alpha]B \wedge [\alpha][\alpha]B \wedge [\alpha][\alpha][\alpha]B \wedge [\alpha][\alpha][\alpha^*]B \tag{11}$$

Using the propositional sequent rules $\to$r and $\wedge$r on (11) leads to

$$\to\text{r}\frac{\wedge\text{r}\frac{A \vdash B \quad \wedge\text{r}\frac{A \vdash [\alpha]B \quad \wedge\text{r}\frac{A \vdash [\alpha][\alpha]B \quad \wedge\text{r}\frac{A \vdash [\alpha][\alpha][\alpha]B \quad A \vdash [\alpha][\alpha][\alpha^*]B}{A \vdash [\alpha][\alpha][\alpha]B \wedge [\alpha][\alpha][\alpha^*]B}}{A \vdash [\alpha][\alpha]B \wedge [\alpha][\alpha][\alpha]B \wedge [\alpha][\alpha][\alpha^*]B}}{A \vdash [\alpha]B \wedge [\alpha][\alpha]B \wedge [\alpha][\alpha][\alpha]B \wedge [\alpha][\alpha][\alpha^*]B}}{A \vdash B \wedge [\alpha]B \wedge [\alpha][\alpha]B \wedge [\alpha][\alpha][\alpha]B \wedge [\alpha][\alpha][\alpha^*]B}}{\vdash A \to B \wedge [\alpha]B \wedge [\alpha][\alpha]B \wedge [\alpha][\alpha][\alpha]B \wedge [\alpha][\alpha][\alpha^*]B}$$

Let us summarize this notationally by the following

$$\to\text{r},\wedge\text{r},\wedge\text{r},\wedge\text{r},\wedge\text{r}\frac{A \vdash B \quad A \vdash [\alpha]B \quad A \vdash [\alpha][\alpha]B \quad A \vdash [\alpha][\alpha][\alpha]B \quad A \vdash [\alpha][\alpha][\alpha^*]B}{\vdash A \to B \wedge [\alpha]B \wedge [\alpha][\alpha]B \wedge [\alpha][\alpha][\alpha]B \wedge [\alpha][\alpha][\alpha^*]B}$$

to recall that there was a derivation involving one use of $\to$r and 4 uses of $\wedge$r from the four premises to the single conclusion without saying which derivation it was exactly. Mentioning $\wedge$r 4 times seems a bit repetitive, so simply abbreviate this as:

$$\to\text{r},\wedge\text{r}\frac{A \vdash B \quad A \vdash [\alpha]B \quad A \vdash [\alpha][\alpha]B \quad A \vdash [\alpha][\alpha][\alpha]B \quad A \vdash [\alpha][\alpha][\alpha^*]B}{\vdash A \to B \wedge [\alpha]B \wedge [\alpha][\alpha]B \wedge [\alpha][\alpha][\alpha]B \wedge [\alpha][\alpha][\alpha^*]B}$$

How could we prove the premises? Sect. 4 investigated one way, which essentially amounts to Bounded Model Checking. Can we be more clever and prove the same premises in a different way? Preferably one that is more efficient and allows us to get the proof over with after finitely many steps?

There is not all that much we can do to improve the way we prove the first premise ($A \vdash B$). We simply have to bite the bullet and do it, armed with all our knowledge of arithmetic from Lecture 6. But it's actually very easy at least for the bouncing ball. Besides, no dynamics have actually happened yet in the first premise, so if we despair in proving this one, the rest cannot become any easier either. For the second premise, there is not much that we can do either, because we will have to analyze the effect of the loop body α running once at least in order to be able to understand what happens if we run α repeatedly.

Yet, what's with the third premise $A \vdash [\alpha][\alpha]B$? We could just approach it as is and try to prove it directly using the dL proof rules. Alternatively, however, we could try to take advantage of the fact that it is the same hybrid program α that is running in the first and the second modality. Maybe they should have something in common that we can exploit as part of our proof?

How could that work? Can we possibly find something that is true after the first run of α and is all we need to know about the state for $[\alpha]B$ to hold? Can we characterize the intermediate state after the first α and before the second α? Suppose we manage to do that and identify a formula E that characterizes the intermediate state in this way. How do we use this intermediate condition E to simplify our proof?

Recall the intermediate condition contract version of the sequential composition proof rule from Lecture 4 on Safety & Contracts that we briefly revisited again in Lecture 5.

$$\text{(R4)}\ \frac{A \to [\alpha]E \quad E \to [\beta]B}{A \to [\alpha;\beta]B}$$

Lecture 5 ended up dismissing the intermediate contract rule R4 in favor of the more general axiom

$$([;])\ [\alpha;\beta]\phi \leftrightarrow [\alpha][\beta]\phi$$

But, let us revisit R4 just the same and see if we can learn something from its way of using intermediate condition E. The first obstacle is that the conclusion of R4 does not match the form we need for $A \vdash [\alpha][\alpha]B$. That's not a problem in principle, because we could use axiom [;] backwards from right-hand side to left-hand side in order to turn $A \vdash [\alpha][\alpha]B$ back into

$$A \vdash [\alpha;\alpha]B$$

and then use rule R4 to generalize with an intermediate condition E in the middle. However, this is what we wanted to stay away from, because using the axioms both forwards and backwards can get our proof search into trouble because we might loop around trying to find a proof forever without making any progress by simply using axiom [;] forwards and then backwards and then forwards again and so on until the end of time. Such a looping proof does not strike us as useful. Instead, we'll adopt a proof rule that has some of the thoughts of R4 but is more general. It is called *generalization* and allows us to prove any stronger postcondition ϕ for a modality, i.e. a postcondition that implies the original postcondition ψ.

Lemma 3 (Generalization rule)**.** *The following is a sound proof rule*

$$([]gen')\ \frac{\Gamma \vdash [\alpha]\phi, \Delta \quad \phi \vdash \psi}{\Gamma \vdash [\alpha]\psi, \Delta}$$

If we apply rule $[]gen'$ on the third premise $A \vdash [\alpha][\alpha]B$ of our bounded model checking style proof attempt with the intermediate condition E for ϕ that we assume to have identified, then we end up with

$$[]gen'\frac{A \vdash [\alpha]E \qquad E \vdash [\alpha]B}{A \vdash [\alpha][\alpha]B}$$

Let us try to use this principle to see if we can find a way to prove

$$A \to B \wedge [\alpha](B \wedge [\alpha](B \wedge [\alpha](B \wedge [\alpha][\alpha^*]B))) \tag{9}$$

Using rules $\wedge$r and $[]gen'$ a number of times for a sequence of intermediate conditions E_1, E_2, E_3 derives:

$$\dfrac{\dfrac{A \vdash B \quad \dfrac{A \vdash [\alpha]E_1 \quad \dfrac{E_1 \vdash B \quad \dfrac{E_1 \vdash [\alpha]E_2 \quad \dfrac{E_2 \vdash B \quad \dfrac{E_2 \vdash [\alpha]E_3 \quad \dfrac{E_3 \vdash B \quad E_3 \vdash [\alpha][\alpha^*]B}{E_3 \vdash B \wedge [\alpha][\alpha^*]B}\wedge r}{E_2 \vdash [\alpha](B \wedge [\alpha][\alpha^*]B)}[]gen'}{E_2 \vdash B \wedge [\alpha](B \wedge [\alpha][\alpha^*]B)}\wedge r}{E_1 \vdash [\alpha](B \wedge [\alpha](B \wedge [\alpha][\alpha^*]B))}[]gen'}{E_1 \vdash B \wedge [\alpha](B \wedge [\alpha](B \wedge [\alpha][\alpha^*]B))}\wedge r}{A \vdash [\alpha](B \wedge [\alpha](B \wedge [\alpha](B \wedge [\alpha][\alpha^*]B)))}[]gen'}{A \vdash B \wedge [\alpha](B \wedge [\alpha](B \wedge [\alpha](B \wedge [\alpha][\alpha^*]B)))}\wedge r}{\vdash A \to B \wedge [\alpha](B \wedge [\alpha](B \wedge [\alpha](B \wedge [\alpha][\alpha^*]B)))}\to r$$

This particular derivation is still not very useful because it still has a loop in one of the premises, which is what we had originally started out with in (5) in the first place. But the derivation hints at a useful way how we could possibly shortcut proofs. To lead to a proof of the conclusion, the above derivation requires us to prove the premises

$$A \vdash [\alpha]E_1$$
$$E_1 \vdash [\alpha]E_2$$
$$E_2 \vdash [\alpha]E_3$$

as well as some other premises. What is an easy case to make that happen? What if all the intermediate conditions E_i were the same? Let's assume they are all the same condition E, that is, $E_1 \equiv E_2 \equiv E_3 \equiv E$. In that case, most of the resulting premises actually turn out to be one and the same premise:

$$E \vdash B$$
$$E \vdash [\alpha]E$$

except for the two left-most and the right-most premise. *Let us leverage this observation and develop a proof rule for which the same intermediate condition is used for all iterates of the loop.* Furthermore, we would even know the first premise

$$A \vdash [\alpha]E$$

if we could prove that the precondition A implies E:

$$A \vdash E$$

because, we already have $E \vdash [\alpha]E$ as one of the premises.

6 Invariant Proofs of Loops

The condition $E \vdash [\alpha]E$ identified in the previous section seems particularly useful, because it basically says that whenever the system α starts in a state satisfying E, it will stay in E, no matter which of the states in E it was where the system started in the first place. It sounds like the system α^* couldn't get out of E either if it starts in E since all

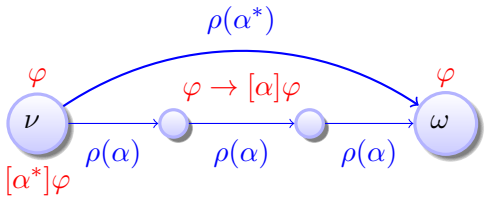
that α^* can do is to repeat α some number of times. But every time we repeat α, the sequent $E \vdash [\alpha]E$ expresses that we cannot leave E that way. So no matter how often our CPS repeats α^*, it will still reside in E.

The other condition that the previous section identified as crucial is $E \vdash B$. And, indeed, if E does not imply the postcondition B that we have been interested in in the first place, then E is a perfectly true invariant of the system, but not a very useful one as far as proving B goes.

What else could go wrong in a system that obeys $E \vdash [\alpha]E$, i.e. where this sequent is valid, because we found a proof for it? Indeed, the other thing that could happen is that E is an invariant of the system that would imply safety, but our system just does not initially start in E, then we still don't know whether it's safe.

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

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



Lemma 4 (Induction). *The (loop) induction rule is sound:*

$$(ind') \frac{\Gamma \vdash \varphi, \Delta \quad \varphi \vdash [\alpha]\varphi \quad \varphi \vdash \psi}{\Gamma \vdash [\alpha^*]\psi, \Delta}$$

First observe that the *inductive invariant* φ (which we called E in the previous more concrete examples) occurs in all premises but not in the conclusion of ind'. That means, whenever we apply the induction rule ind' to a desired conclusion, we get to choose what invariant φ we want to use it for. Good choices of φ will lead to a successful proof of the conclusion. Bad choices of φ will stall the proof, because some of the premises cannot be proved.

The first premise of ind' says that the initial state, about which we assume Γ (and that Δ does not hold), satisfies the invariant φ, i.e. the invariant is initially true. The second premise of ind' shows that the invariant φ is *inductive*. That is, whenever φ was true before running the loop body α, then φ is always true again after running α. The third premise of ind' shows that the invariant φ is strong enough to imply the postcondition ψ that the conclusion was interested in.

Rule ind' says that ψ holds after any number of repetitions of α if an invariant φ holds initially (left premise), if invariant φ remains true after one iteration of α from any state where φ was true (middle premise), and if invariant φ finally implies the

desired postcondition ψ (right premise). If φ is true after executing α whenever φ has been true before (middle premise), then, if φ holds in the beginning (left premise), φ will continue to hold, no matter how often we repeat α in $[\alpha^*]\psi$, which is enough to imply $[\alpha^*]\psi$ if φ implies ψ (right premise).

Taking a step back, these three premises look somewhat familiar, because they correspond exactly to the proof steps that the 15-122 Principles of Imperative Computation course used to show that the contract of a function with a `@requires` contract Γ (and not Δ), `@ensures` contract ψ, and a loop invariant φ is correct. Now, we have this reasoning in a more general and formally more precisely defined context. We no longer need to appeal to intuition to justify why such a proof rule is fine, but can evoke a soundness proof for ind'. We will also no longer be limited to informal arguments to justify invariance for a program but can do actual solid and rigorous formal proofs if we combine proof rule ind' with the other proof rules from Lecture 6 on Truth & Proof.

7 A Proof of a Repetitive Bouncing Ball

Now that it understand the principles of how to prove loops in CPS, the bouncing ball is eager to put these skills to use. The ball wants to relieve itself of its acrophobic fears once and for all by conducting a proof that it won't ever have to be afraid of excess heights $> H$ again nor of falling through the cracks in the ground to heights < 0.

The bouncing ball again use its favorite abbreviations:

$$\begin{aligned} A_{x,v} &\overset{\text{def}}{\equiv} 0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 \geq c \geq 0 \\ B_{x,v} &\overset{\text{def}}{\equiv} 0 \leq x \land x \leq H \\ (x'' = \dots) &\overset{\text{def}}{\equiv} (x' = v, v' = -g \,\&\, x \geq 0) \end{aligned}$$

Note the somewhat odd abbreviation for the differential equation just to simplify notation, so that the bouncing ball conjecture (3) is:

$$A_{x,v} \to [(x' = \dots; (?x = 0; v := -cv \cup ?x > 0)^*]B_{x,v} \tag{3}$$

The first thing that the bouncing ball will need for the proof of (3) is the appropriate choice for the invariant φ to be used in the induction proof rule ind'. The bouncing ball will use a dL formula $E_{x,v}$ for the invariant when instantiating φ in the proof rule ind'. But the ball is still a little unsure about how exactly to define that formula $E_{x,v}$, not an unusual situation when trying to master the understanding of CPS. Can you think of a good choice for the formula $E_{x,v}$ to help the bouncing ball?

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

I don't know about you, but the bouncing ball settles for the choice of using its post-condition as an invariant, because that is what it wants to show about its behavior:

$$E_{x,v} \overset{\text{def}}{\equiv} 0 \leq x \wedge x \leq H \tag{12}$$

Because the bouncing ball is so proud of its wonderful invariant $E_{x,v}$, it even uses it to perform a generalization with the newly acquired skill of the generalization proof rule $[]gen'$ in the inductive step to completely separate the proof about the differential equation and the proof about the bouncing dynamics.[1] Let there be proof.

$$\dfrac{A_{x,v} \vdash E_{x,v} \quad \dfrac{E_{x,v} \vdash [x' = \ldots]E_{x,v} \quad []gen' \dfrac{\dfrac{\dfrac{\dfrac{E_{x,v}, x = 0 \vdash E_{x,-cv}}{E_{x,v}, x = 0 \vdash [v := -cv]E_{x,v}}[:=]\text{r}}{E_{x,v} \vdash [?x = 0][v := -cv]E_{x,v}}[?]\text{r}}{E_{x,v} \vdash [?x = 0; v := -cv]E_{x,v}}[;]\text{r} \quad \dfrac{E_{x,v}, x > 0 \vdash E_{x,v}}{E_{x,v} \vdash [?x > 0]E_{x,v}}[?]\text{r}}{E_{x,v} \vdash [?x = 0; v := -cv]E_{x,v} \wedge [?x > 0]E_{x,v}}\wedge\text{r}}{E_{x,v} \vdash [?x = 0; v := -cv \cup ?x > 0]E_{x,v}}[\cup]\text{r}}{E_{x,v} \vdash [x' = \ldots][?x = 0; v := -cv \cup ?x > 0]E_{x,v}}[;]\text{r}}{A_{x,v} \vdash [(x' = \ldots; (?x = 0; v := -cv \cup ?x > 0)^*]B_{x,v}}\to\text{r}}{\vdash A_{x,v} \to [(x' = \ldots; (?x = 0; v := -cv \cup ?x > 0)^*]B_{x,v}}\,ind' \quad E_{x,v} \vdash B_{x,v}$$

This sequent proof has 5 premises remaining to be proved. The bouncing ball is pretty sure how to prove the first premise ($A_{x,v} \vdash E_{x,v}$) corresponding to the initial condition, because $0 \leq x \leq H$ is true initially as $0 \leq x = H$ follows from $A_{x,v}$. The bouncing ball also knows how to prove the last premise ($E_{x,v} \vdash B_{x,v}$), because the invariant $E_{x,v}$ from (12) happens to be equal to the desired postcondition $B_{x,v}$, so this holds by axiom ax. But the bouncing ball is running into unforeseen(?) trouble with the inductive step in the middle. While the third and fourth premise hold, the second premise $E_{x,v} \vdash [x' = \ldots]E_{x,v}$ with the differential equation resists a proof for the choice (12). And that makes sense, because, even if the current height is bounded by $0 \leq x \leq H$ before the differential equation, there is no reason to believe it would remain bounded afterwards if this is all we know about the bouncing ball. After all, if the ball is just below $x = H$, it would still ultimately exceed H if its velocity were too big.

Ah, right! We actually found that out about the bouncing ball in Lecture 4 on Safety & Contracts already when were were wondering under which circumstance it might be safe to let a ball bounce around. And, as a matter of fact, everything we learn by Principle of Cartesian Doubt about when it would be safe to start a CPS can be very valuable information to preserve in the invariant. If it wasn't safe to start a CPS in a state, chances are, it wouldn't be safe either if we kept it running in such a state as we do in an inductive step.

Well, so the ball found a (poor) choice of an invariant $E_{x,v}$ as in (12) that just does not prove because of the inductive step. What to do wonders our little bouncing ball.

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

[1] This is not necessary and the bouncing ball might just as well not have used $[]gen'$ and go for a direct proof using $[']\text{r}$ right away instead. But it does save us some space on this page if the bouncing ball goes for that and also serves as a showcase for the practical use of proof rule $[]gen'$.

There was trouble in the induction step, because $x \leq H$ could not be proved to be inductive. So the bouncing ball could demand a little less from the invariant and use the following weaker choice for $E_{x,v}$ instead of (12):

$$E_{x,v} \overset{\text{def}}{\equiv} x \geq 0 \tag{13}$$

Armed with this new choice for an invariant, the bouncing ball quickly gets to work constructing a new proof for (3). After frantically scribbling a couple of pages with sequent proofs, the bouncing ball experiences a *déjà vu* and notices that its new proof has exactly the same form as the last sequent proof it began. Just with a different choice for the logical formula $E_{x,v}$ to be used as the invariant when applying rule ind'. With the choice (13) rather than (12). Fortunately, the bouncing ball already worked with an abbreviation last time it started a proof, so it is actually not surprising after all to see that the proof structure stays exactly the same and that the particular choice of $E_{x,v}$ only affects the premises, not the way the proof unraveled its program statements in the modalities.

Inspecting the 5 premises of the above sequent proof attempt in light of the improved choice (13) for the invariant, the bouncing ball is delighted to find out that the inductive step works out just fine. The height stays above ground always by construction with the evolution domain constraint $x \geq 0$ and is not changed in the subsequent discrete bouncing control. The initial condition ($A_{x,v} \vdash E_{x,v}$) also works out alright, because $0 \leq x$ was among the assumptions in $A_{x,v}$. Only this time, the last premise ($E_{x,v} \vdash B_{x,v}$) falls apart, because $x \geq 0$ is not at all enough to conclude the part $x \leq H$ of the post-condition. What's a ball to do to get itself verified these days?

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

The bouncing ball takes the lesson from Cartesian Doubt to heart and realizes that the invariant needs to transport enough information about the state of the system to make sure the inductive step has a chance of holding true. In particular, the invariant desperately needs to preserve knowledge about the velocity, because how the height changes depends on the velocity (after all the differential equation reads $x' = v, \ldots$), so it would be hard to get a handle on height x without first understanding how velocity v changes.

Fine, so the bouncing ball quickly discards the failed invariant choice from (12), which it is no longer so proud of, and also gives up on the weaker version (13), but instead shoots for a stronger invariant of which it would be sure to be inductive and strong enough to imply safety:

$$E_{x,v} \overset{\text{def}}{\equiv} x = 0 \land v = 0 \tag{14}$$

This time, the bouncing ball learned its lesson and won't blindly set out to prove the property (3) from scratch again, but, rather, be clever about it and realize that it's still going to find the same shape of the sequent proof attempt above, just with a, once again, different choice for the invariant $E_{x,v}$. So the bouncing ball quickly jumps to conclusions and inspects its famous 5 premises of the above sequent proof attempt. This time, the postcondition works out easily and the inductive step works like a charm (no velocity, no height, no motion). But the initial condition is giving it a headache, because there is no reason to believe the ball would initially lie flat on the ground with velocity zero.

For a moment there, the bouncing ball fancied the option of simply changing the initial condition $A_{x,v}$ around to include $x = 0$, because that would make this proof attempt work out swell. But then it realized that this would mean the bouncing ball would from now on be doomed to only start the day at speed zero on the ground, which would not lead to all that much excitement for a cheerful bouncing ball. That would be safe, but a bit too much so for lack of motion.

What, then, is the bouncing ball supposed to do to finally get a proof without crippling its permitted initial conditions?

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

This time, the bouncing ball thinks real hard and has a smart idea. Thinking back of how the lecture notes had motivated the idea of invariants for loops, commonalities of states before and after running the loop body as well as intermediate conditions featured a prominent role in shaping the intuition for invariants. Lecture 4 on Safety & Contracts had already identified an intermediate condition for the single-hop bouncing ball. Maybe that will prove useful as an invariant, too:

$$E_{x,v} \overset{\text{def}}{\equiv} 2gx = 2gH - v^2 \land x \geq 0 \tag{15}$$

After all, an invariant is something like a permanent intermediate condition, i.e. an intermediate condition that keeps on working out alright for all future iterations. The bouncing ball is not so sure whether this will work but it seems worth trying.

The shape of the above proof again stays exactly the same, just with a different choice of $E_{x,v}$, this time coming from (15). The remaining 5 premises are then proved easily. The first premise $A_{x,v} \vdash E_{x,v}$ proves easily using $x = H$ and $v = 0$:

$$0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 \geq c \geq 0 \vdash 2gx = 2gH - v^2 \land x \geq 0$$

Recalling the usual abbreviations, the second premise $E_{x,v} \vdash [x' = \ldots]E_{x,v}$ is

$$2gx = 2gH - v^2 \land x \geq 0 \vdash [x' = v, v' = -g \,\&\, x \geq 0](2gx = 2gH - v^2 \land x \geq 0)$$

a proof whose pieces we have seen in previous lectures (Exercise 2). The third premise $E_{x,v}, x = 0 \vdash E_{x,-cv}$ is

$$2gx = 2gH - v^2 \land x \geq 0, x = 0 \vdash 2gx = 2gH - (-cv)^2 \land x \geq 0$$

which would prove easily if we knew $c = 1$. Do we know $c = 1$? No, we do not know $c = 1$, because we only assumed $1 \geq c \geq 0$ in $A_{x,v}$. But we could prove this third premise easily if we would also change the definition of the initial condition $A_{x,v}$ around to include $c = 1$. That may not be the most general possible statement about bouncing balls, but let's happily settle for it. Note that even then, however, we still need to augment $E_{x,v}$ to include $c = 1$ as well, since we otherwise would have lost this knowledge before we need it in the third premise. Having lost critical pieces of knowledge is a phenomenon you may encounter when you are conducting proofs. In that case, you should trace where you lost the assumption in the first place and put it back in. But then you have also learned something valuable about your system, namely which assumptions are crucial for the correct functioning of which part of the system. The fourth premise, $E_{x,v}, x \geq 0 \vdash E_{x,v}$ proves whatever the abbreviations stand for simply using the axiom rule ax. In fact, the bouncing ball could have noticed this earlier already but might have been distracted by its search for a good choice for the invariant $E_{x,v}$. This is but one indication for the fact that it may pay off to take a step back from a proving effort and critically reflect on what all the pieces of the argument rely on exactly. Finally, the fifth premise $E_{x,v} \vdash B_{x,v}$, which is

$$2gx = 2gH - v^2 \land x \geq 0 \vdash 0 \leq x \land x \leq H$$

proves easily with arithmetic as long as we know $g > 0$. This condition is already included in $A_{x,v}$. But we still managed to forget about that in our invariant $E_{x,v}$. So, again, $g > 0$ should have been included in the invariant $E_{x,v}$, which, overall, should have been defined as

$$E_{x,v} \overset{\text{def}}{\equiv} 2gx = 2gH - v^2 \land x \geq 0 \land c = 1 \land g > 0 \tag{16}$$

This is nearly the same definition as (15) except that assumptions about the system parameter choices are carried through. The last two conjuncts are trivial, because neither c nor g changes while the little bouncing ball falls. We, unfortunately, still have to include it in the invariant. This is one of the downsides of working with intermediate condition style proofs such as what we get with rule $[]gen'$. Later lectures investigate significant simplifications for this nuisance and will enable you to elide the trivial constant part $c = 1 \land g > 0$ from the invariant.

For the record, we now really have a full sequent proof of the undamped bouncing ball with repetitions. Exciting!

$$\begin{aligned} &0 \leq x \land x = H \land v = 0 \land g > 0 \land 1 = c \rightarrow \\ &\quad [\big(x' = v, v' = -g \,\&\, x \geq 0; (?x = 0; v := -cv \cup ?x \geq 0)\big)^*](0 \leq x \land x \leq H) \end{aligned} \tag{17}$$

Since invariants are a crucial part of a CPS design, you are encouraged to always describe invariants in your hybrid programs: Let's capture the bouncing ball's contracts, which has now been verified by way of proving the corresponding dℒ formula (17):

$$\begin{aligned} &\texttt{@requires}(0 \leq x \land x = H \land v = 0) \\ &\texttt{@requires}(g > 0 \land c = 1) \\ &\texttt{@ensures}(0 \leq x \land x \leq H) \\ &\big(x' = v, v' = -g \,\&\, x \geq 0; \\ &\quad (?x = 0; v := -cv \cup ?x \geq 0)\big)^*\texttt{@invariant}(2gx = 2gH - v^2 \land x \geq 0 \land c = 1 \land g > 0) \end{aligned} \tag{18}$$

KeYmaera will also make use of the invariants annotated using the `@invariant` contract in hybrid programs to simplify your proof effort. But KeYmaera does not require a list of the constant expressions in the invariant contracts. So the following slight simplification of (18) will suffice:

$$\begin{aligned} &\texttt{@requires}(0 \leq x \land x = H \land v = 0) \\ &\texttt{@requires}(g > 0 \land c = 1) \\ &\texttt{@ensures}(0 \leq x \land x \leq H) \\ &\big(x' = v, v' = -g \,\&\, x \geq 0; \\ &\quad (?x = 0; v := -cv \cup ?x \geq 0)\big)^*\texttt{@invariant}(2gx = 2gH - v^2 \land x \geq 0) \end{aligned}$$

Since KeYmaera uses `@invariant` contracts whenever possible, it is a good idea to rephrase (17) by explicitly including the invariant contract as:

$$\begin{array}{l} 0 \leq x \wedge x = H \wedge v = 0 \wedge g > 0 \wedge 1 = c \rightarrow \\ \quad [\big(x' = v, v' = -g \,\&\, x \geq 0; \\ \quad\ (?x = 0; v := -cv \cup ?x \geq 0)\big)^* \texttt{@invariant}(2gx = 2gH - v^2 \wedge x \geq 0)] \\ \quad (0 \leq x \wedge x \leq H) \end{array} \tag{19}$$

8 Essentials of Induction & Cuts

The induction rule ind' is very useful in practice. But there is also a more elegant and more essential way of stating the induction principle.

Lemma 5 (Induction). *The induction rule is sound:*

$$(ind)\ \frac{\varphi \vdash [\alpha]\varphi}{\varphi \vdash [\alpha^*]\varphi}$$

The new rule ind is clearly a special case of rule ind', obtained by specializing $\Gamma \overset{\text{def}}{\equiv} \psi$, $\Delta = .$ (empty), and $\varphi \overset{\text{def}}{\equiv} \psi$, in which case the left and right premises of ind' are provable directly by ax so that only the middle premise remains. If ind is a special case of ind', why should we still prefer ind from a perspective of essentials? Obviously, ind is more fundamental and easier. But if this came at the cost of being less powerful, ind' should still be preferred. It turns out that ind' is actually a special case of ind with a little extra work. This extra work needs a bit of attention but is insightful.

Let's adopt the following variation of the generalization rule:

$$([]gen)\ \frac{\phi \vdash \psi}{[\alpha]\phi \vdash [\alpha]\psi}$$

For example, using a cut with $\varphi \rightarrow [\alpha^*]\varphi$, rule ind' can be derived from ind and $[]gen$ as follows (using weakening WI,Wr without notice):

$$\cfrac{\cfrac{\cfrac{\varphi \vdash [\alpha]\varphi}{\varphi \vdash [\alpha^*]\varphi}\,ind}{\Gamma \vdash \varphi \rightarrow [\alpha^*]\varphi, \Delta}\,{\rightarrow}\text{r} \qquad \cfrac{\Gamma \vdash \varphi, \Delta \qquad \cfrac{\varphi \vdash \psi}{[\alpha^*]\varphi \vdash [\alpha^*]\psi}\,[]gen}{\Gamma, \varphi \rightarrow [\alpha^*]\varphi \vdash [\alpha^*]\psi, \Delta}\,{\rightarrow}\text{l}}{\Gamma \vdash [\alpha^*]\psi, \Delta}\,cut$$

Hence ind' is a derived rule, because it can be derived using ind and some other rules. Thus, ind' is not necessary in theory, but still useful in practice.

Yet, now, in order to derive rule ind' out of the more fundamental ind, we had to add the revised generalization rule $[]gen$. Is that any easier? Well it is, because $[]gen$ actually makes $[]gen'$ unnecessary by another smart argument using a cut with the desired formula $[\alpha]\phi$.

$$\cut\dfrac{\dfrac{\Gamma \vdash [\alpha]\phi, \Delta}{\Wr \Gamma \vdash [\alpha]\phi, [\alpha]\psi, \Delta} \qquad \Wl,\Wr\dfrac{\dfrac{\phi \vdash \psi}{\gbox gen\ [\alpha]\phi \vdash [\alpha]\psi}}{\Gamma, [\alpha]\phi \vdash [\alpha]\psi, \Delta}}{\Gamma \vdash [\alpha]\psi, \Delta}$$

This leaves exactly the premises of rule $\gbox gen'$, making $\gbox gen'$ a derived rule. Whenever we need $\gbox gen'$, we could simply expand the proof out in the above form to reduce it just a proof involving $\gbox gen$ and *cut* and weakening.

These are two illustrations how creative uses of cuts can suddenly make proves and concepts easier. A phenomenon that we will see in action much more often in this course.

Before you despair that you would have to derive ind' and $\gbox gen'$ every time you need them: that is not the case. The theorem prover KeYmaera is very well aware of how useful both versions of the proof rules are and has them at your disposal. For theoretical investigations, however, as well as for understanding the truly fundamental reasoning steps, it is instructive to see that ind and $\gbox gen$ are fundamental, while the others are mere consequences.

9 Summary

This lecture focused on developing and using the concept of invariants for CPS. Invariants enable us to prove properties of CPS with loops, a problem of ubiquitous significance, because hardly any CPS get by without repeating some operations in a control loop. Invariants constitute the single most insightful and most important piece of information about a CPS.

Note 7. *This lecture discussed a number of useful proof rules, including:*

$$(ind')\ \frac{\Gamma \vdash \varphi, \Delta \quad \varphi \vdash [\alpha]\varphi \quad \varphi \vdash \psi}{\Gamma \vdash [\alpha^*]\psi, \Delta}$$

$$(\gbox\wedge r)\ \frac{\Gamma \vdash [\alpha]B \wedge [\alpha]\psi, \Delta}{\Gamma \vdash [\alpha](B \wedge \psi), \Delta}$$

$$(\gbox\wedge l)\ \frac{\Gamma, [\alpha]B \wedge [\alpha]\psi \vdash \Delta}{\Gamma, [\alpha](B \wedge \psi) \vdash \Delta}$$

$$(\gbox gen')\ \frac{\Gamma \vdash [\alpha]\phi, \Delta \quad \phi \vdash \psi}{\Gamma \vdash [\alpha]\psi, \Delta}$$

The development that led to invariants has some interesting further consequences especially for finding bugs in CPS by unrolling loops and disproving the resulting premises. But this bounded model checking principle is of limited use for ultimately verifying safety, because it only considers the system some finite number of steps in the future. You may find unwinding useful when you are looking for bugs in your CPS, though.

In our effort of helping the bouncing ball succeed with its proof, we saw a range of reasons why an inductive proof may not work out and what needs to be done to adapt the invariant.

Exercises

Exercise 1 (Give bouncing ball back its evolution domain). Explain why the transformation from (3) to (4) was okay in this case.

Exercise 2. Give a sequent proof for

$$2gx = 2gH - v^2 \land x \geq 0 \rightarrow [x' = v, v' = -g \,\&\, x \geq 0](2gx = 2gH - v^2 \land x \geq 0)$$

Does this property also hold if we remove the evolution domain constraint $x \geq 0$? That is, is the following formula valid?

$$2gx = 2gH - v^2 \land x \geq 0 \rightarrow [x' = v, v' = -g](2gx = 2gH - v^2 \land x \geq 0)$$

Exercise 3. To develop an inductive proof rule, we have started systematic unwinding considerations from formula (9) in Sect. 5. In lecture, we started from the form (11) instead and have seen that that takes us to the same inductive principle. Which of the two ways of proceeding is more efficient? Which one produces less premises that are distractions in the argument? Which one has less choices of different intermediate conditions E_i in the first place?

Exercise 4. Could the bouncing ball use any of these formulas as invariants to prove (3)?

$$E_{x,v} \overset{\text{def}}{\equiv} (x = 0 \lor x = H) \land v = 0$$
$$E_{x,v} \overset{\text{def}}{\equiv} 0 \leq x \land x \leq H \land v^2 \leq 2gH$$
$$E_{x,v} \overset{\text{def}}{\equiv} 0 \leq x \land x \leq H \land v \leq 0$$

Exercise 5. Conduct a sequent proof for (17) without using the generalization rule $[]gen'$.

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.

[PCL11] Frank Pfenning, Thomas J. Cortina, and William Lovas. Teaching imperative programming with contracts at the freshmen level. 2011.

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

[Pla12b] André Platzer. Logics of dynamical systems. In *LICS*, pages 13–24. IEEE, 2012. doi:10.1109/LICS.2012.13.