15-424: Foundations of Cyber-Physical Systems

Lecture Notes on Hybrid Systems & Games

André Platzer

Carnegie Mellon University
Lecture 16

1 Introduction

Hybrid systems have so far served us well throughout this course as a model for cyber-physical systems [Pla08, Pla10b, Pla12c]. Most definitely, hybrid systems can also serve as models for other systems that are not cyber-physical per se, i.e. they are not built as a combination of cyber and computing capabilities with physical capabilities. Some biological systems can be understood as hybrid systems, because they combine discrete and continuous dynamics. Or physical processes in which things happen at very different speeds, so where there is a slow process about which a continuous understanding is critical as well as a very fast process in which a discrete abstraction might be sufficient. Neither of those examples are particularly cyber-physical. Yet, nevertheless, they can have natural models as hybrid systems, because their fundamental characteristics is the interaction of discrete and continuous dynamics, which is exactly what hybrid systems are good for. Hence, despite their good match, not *all* hybrid systems are cyber-physical systems.

One important point of today's lecture is that the converse is not true either. Not all cyber-physical systems are hybrid systems. The reason for that is *not* that cyber-physical systems lack discrete and continuous dynamics, but, rather, that they involve also additional dynamical aspects. It is a common phenomenon in cyber-physical systems that they involve several dynamical aspects, which is why they are best understood as *multi-dynamical systems*, i.e. systems with multiple dynamical features [Pla12c, Pla12b, Pla11, Pla15].

In a certain sense, applications often have a $+1$ effect on dynamical aspects. Your analysis might start out focusing on some number of dynamical aspects just to observe during the elaboration of the analysis that there is a part of the system for which one more dynamical aspect is relevant than was originally anticipated. The bouncing ball

is an example where a preliminary analysis might first ascribe an entirely continuous dynamics to it, just to find out after a while that the singularity of bouncing back from the ground would be better understood by discrete dynamics. So whenever you are analyzing a system, be prepared to find one more dynamical aspect around the corner. That is yet another reason why it is useful to have flexible and general analysis techniques grounded in logic that still work even after a new dynamical aspect has been found.

Of course, it is not going to be feasible to understand all multi-dynamical system aspects at once in today's lecture. But today's lecture is going to introduce one very fundamental dynamical aspect: *adversarial dynamics* [Pla15]. Adversarial dynamics comes from multiple players that, in the context of CPS, interact on a hybrid system and are allowed to make their respective choices arbitrarily, just in pursuit of their goals. The combination of discrete, continuous, and adversarial dynamics leads to *hybrid games*. Unlike hybrid systems, hybrid games allow choices in the system dynamics to be resolved adversarially by different players with different objectives.

Hybrid games are necessary in situations where multiple agents actively compete. The canonical situation of a hybrid game would, thus, be RoboCup, where two teams of robots play robot soccer, moving around physically in space, controlled according to discrete computer decisions, and in active competition for scoring goals in opposite directions on the field. The robots in a RoboCup match just can't agree on the direction into which they try to get the ball rolling. It turns out, however, that hybrid games also come up for reasons of analytic competition, that is, where possible competition is assumed only for the sake of a worst-case analysis.

Consider lab 4, the static and dynamic obstacles lab, for example, where your robot is interacting with a roguebot. You are in control of the robot, but somebody else is controlling the roguebot. Your objective is to control your robot so that it will not run into the roguebot no matter what. That means you need to find *some* way of playing your control choices for your robot so that it makes progress but will be safe *for all* possible control choices that the roguebot might follow. After all you do not exactly know how the other roguebot is implemented and how it will react to your control decisions. That makes your robot play a hybrid game with the roguebot in which your robot is trying to safely avoid collisions. The roguebot might behave sanely and tries to stay safe as well. But the roguebot's objectives could differ from yours, because its objective is not to get you to your goal. The roguebot rather wants to get to its own goal instead, which might cause unsafe interferences whenever the roguebot takes an action in pursuit of its goal that is not in your robot's interest. If your robot causes a collision, because it chose an action that was incompatible with the roguebot's action, your robot would certainly be faulty and sent back to the design table.

Alas, when you try to understand how you need to control your robot to stay safe, it can be instructive to think about what the worst-case action of a roguebot might be to make life difficult for you. And when your friendly course instructors try to demonstrate for you under which circumstance a simulation of your robot controller exhibits a faulty behavior, so that you can learn from the cases where your control does not work, they might actually be playing a hybrid game with you. If your robot wins

and stays safe, that is an indication of a strong robot design. But if your course TAs win and show an unsafe trace, you still win even if you lose this particular simulation, because you learn more about the corner cases in your robot control design than when staring at simulation movies where everything is just fair-weather control.

If you think carefully again about lab 2, where your robot was put on a highway and had to find some way of being controlled to stay safe for all possible choices of the robot in front of it, then you will find that a hybrid game interpretation might be in order for that lab as well.

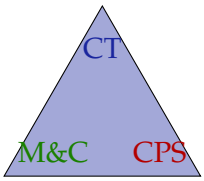
These lecture notes are based on [Pla15], where more information can be found on logic and hybrid games. The most important learning goals of this lecture are:

Modeling and Control: We identify an important additional dynamical aspect, the aspect of *adversarial dynamics*, which adds an adversarial way of resolving the choices in the system dynamics. This dynamical aspect is important for understanding the core principles behind CPS, because multiple agents with possibly conflicting actions are featured frequently in CPS applications. It is helpful to learn under which circumstance adversarial dynamics is important for understanding a CPS and when in can be neglected without loss. CPS in which all choices are resolved against you or all choices are resolved for you can already be described and analyzed in differential dynamic logic. Adversarial dynamics is interesting in mixed cases, where some choices fall in your favor and others turn out against you. Another important goal of this lecture is how to develop models and controls of CPS with adversarial dynamics corresponding to multiple agents.

Computational Thinking: This lecture follows fundamental principles from computational thinking to capture the new phenomenon of adversarial dynamics in CPS models. We leverage core ideas from programming languages by extending syntax and semantics of program models and specification and verification logics with the complementary operator of duality to incorporate adversariality in a modular way into the realm of hybrid systems models. This leads to a compositional model of hybrid games with compositional operators. Modularity makes it possible to generalize our rigorous reasoning principles for CPS to hybrid games while simultaneously taming their complexity. This lecture introduces *differential game logic* dGL [Pla15] extending by adversarial dynamics the familiar differential dynamic logic, which has been used as the specification and verification language for CPS in the other parts of this course. Computer science ultimately is about analysis like worst-case analysis or expected-case analysis or correctness analysis. Hybrid games enable analysis of CPS at a more fine-grained level in between worst-case analysis and best-case analysis. In the dL formula $[\alpha]\phi$ all choices are resolved against us in the sense that $[\alpha]\phi$ is only true if ϕ holds after all runs of α. In the dL formula $\langle\alpha\rangle\phi$ all choices are resolved in favor in the sense that $\langle\alpha\rangle\phi$ is true if ϕ holds after at least one run of α. Hybrid games can be used to attribute some but not all of the choices in a system to an opponent while leaving the others to be resolved favorably. Finally, this lecture provides a perspective on advanced models of computation with alternating choices.

CPS Skills: We add a new dimension into our understanding of the semantics of a CPS model: the adversarial dimension corresponding to how a system changes state over time as multiple agents react to each other. This understanding is crucial for developing an intuition for the operational effects of multi-agent CPS. The presence of adversarial dynamics will cause us to reconsider the semantics of CPS models to incorporate the effects of multiple agents and their mutual reactions. This generalization, while crucial for understanding adversarial dynamics in CPS, also shines a helpful complementary light on the semantics of hybrid systems without adversariality by causing us to reflect on choices.

fundamental principles of computational thinking
logical extensions
PL modularity principles
compositional extensions
differential game logic
best-worst-case analysis
models of alternating computation



adversarial dynamics
conflicting actions
multi-agent systems
angelic/demonic choice

multi-agent state change
CPS semantics
reflections on choices

2 Choices & Nondeterminism

Note 1 (Choices in hybrid systems)**.** *Hybrid systems involve choices. They manifest evidently in hybrid programs as nondeterministic choices $\alpha \cup \beta$ whether to run HP α or HP β, in nondeterministic repetitions α^* where the choice is how often to repeat α, and in differential equations $x' = f(x)\,\&\,Q$ where the choice is how long to follow that differential equation. All those choices, however, have still been resolved in one way, i.e. by the same entity or player.*

In which way the various choices are resolved depends on the context. In the box modality $[\alpha]$ of differential dynamic logic [Pla08, Pla10b, Pla12c], the choices are resolved in *all possible ways* so that the modal formula $[\alpha]\phi$ expresses that formula ϕ holds for all ways how the choices in HP α could resolve. In the diamond modality $\langle\alpha\rangle$, instead, the choices are resolved in *some way* so that formula $\langle\alpha\rangle\phi$ expresses that formula ϕ holds for one way of resolving the choices in HP α. That is how $[\alpha]\phi$ expresses that ϕ holds necessarily after α while $\langle\alpha\rangle\phi$ expresses that ϕ is possible after α.

In particular, choices in α help $\langle\alpha\rangle\phi$, because what this formulas calls for is *some* way of making ϕ happen after α. If α has many possible behaviors, this is easier to satisfy. Choices in α hurt $[\alpha]\phi$, however, because this formula requires ϕ to hold for all those choices. The more choices there are, the more difficult it is to make sure that ϕ holds after every single combination of those choices.

Note 2. *In differential dynamic logic, choices in α either help uniformly (when they occur in $\langle\alpha\rangle\phi$) or make things more difficult uniformly (when they occur in $[\alpha]\phi$).*

That is why these various forms of choices in hybrid programs have been called *nondeterministic*. They are "unbiased". All possible resolutions of the choices in α could happen nondeterministically when running α. Which possibilities we care about (all or some) just depends on the modal formula around it.

3 Control & Dual Control

Another way of looking at the choices that are to be resolved during the runs of a hybrid program α is that they can be resolved by one player. Let's call her *Angel*, because she helps us so much in making $\langle\alpha\rangle\phi$ formulas true. Whenever a choice is about to happen (by running the program statements $\alpha \cup \beta$, α^*, or $x' = f(x)\,\&\,Q$), Angel is called upon to see how the choice is supposed to be resolved this time.

From that perspective, it sounds easy enough to add a second player. Let's call him *Demon* as Angel's perpetual opponent.[1] Only so far, Demon will probably be rather bored after a while, when he realizes that he never actually gets to decide anything, because Angel has all the fun in choosing how the hybrid program world unfolds and Demon just sits around idly. So to keep Demon entertained, we need to introduce some choices that fall under Demon's control.

One thing, we could do to keep Demon interested in playing along is to add a pair of shiny new controls especially for him. They might be called $\alpha \cap \beta$ for Demon's choice between α or β as well as $\alpha^\times$ for repetition of α under Demon's control as well as an operation, say $x' = f(x)\,\&\,Q^d$, for continuous evolution under Demon's reign. But that would cause a lot of attention to Demon's control, which might make him feel overly majestic. Let's not do that, because we don't want Demon to get any ideas.

Instead, we will find it sufficient to add just a single operator to hybrid programs: the dual operator ^{d}. What α^d does is to give all control that Angel had in α to Demon, and, vice versa, all control that Demon had in α to Angel. The dual operator, thus, is a little bit like what happens when you turn a chessboard around by 180° in the middle of the game. Whoever played the choices of player White before suddenly controls Black, and whoever played Black now controls White. With just this single duality operator it turns out that Demon still gets his own set of controls ($\alpha \cap \beta$, $\alpha^\times$, $x' = f(x)\,\&\,Q^d$) by suitably nesting the operators, but we did not have to give him those

[1] The responsibilities of such ontologically loaded names are easier to remember than those of neutral player names I and II.

controls specifically. Yet, now those extra controls are not special but simply an aspect of a more fundamental principle: duality.

4 Hybrid Games

Differential game logic (dGℒ) is a logic for studying properties of hybrid games. The idea is to describe the game form, i.e. rules, dynamics, and choices of the particular hybrid game of interest, using a program notation and to then study its properties by proving the validity of logical formulas that refer to the existence of winning strategies for objectives of those hybrid games. Even though hybrid game forms only describe the game *form* with its dynamics and rules and choices, not the actual objective, they are still simply called hybrid games. The objective for a hybrid game is defined in the modal logical formula that refers to that hybrid game form.

Definition 1 (Hybrid games)**.** The *hybrid games of differential game logic* dGℒ are defined by the following grammar (α, β are hybrid games, x a vector of variables, $f(x)$ a vector of (polynomial) terms of the same dimension, Q is a dGℒ formula or just a formula of first-order real arithmetic):

$$\alpha, \beta \ ::= \ x := e \mid x' = f(x) \,\&\, Q \mid ?Q \mid \alpha \cup \beta \mid \alpha; \beta \mid \alpha^* \mid \boldsymbol{\alpha^d}$$

The only syntactical difference of hybrid games compared to hybrid programs for hybrid systems from Lecture 3 on Choice & Control is that, unlike hybrid programs, hybrid games allow the dual operator α^d. This minor syntactic change also requires us to reinterpret the meaning of the other operators in a much more flexible way to make sense of the presence of subgames within the games in which the players already interact. The basic principle is that whenever there used to be nondeterminism in the hybrid program semantics, there will now be a choice of Angel in the hybrid game semantics. But don't be fooled. The parts of such a hybrid game may still be hybrid games, in which players interact, rather than just a single system running. So *all* operators of hybrid games still need a careful understanding as games, not just $\cdot^d$, because all operators can be applied to subgames that mention $\cdot^d$ or be part of a context that mentions $\cdot^d$.

The *atomic games* of dGℒ are assignments, continuous evolutions, and tests. In the *deterministic assignment game* (or discrete assignment game) $x := \theta$, the value of variable x changes instantly and deterministically to that of θ by a discrete jump without any choices to resolve. In the *continuous evolution game* (or continuous game) $x' = f(x) \,\&\, Q$, the system follows the differential equation $x' = f(x)$ where the duration is Angel's choice, but Angel is not allowed to choose a duration that would, at any time, take the state outside the region where formula Q holds. In particular, Angel is deadlocked and loses immediately if Q does not hold in the current state, because she cannot even evolve for duration 0 then without being outside Q.[2] The *test game* or *challenge* $?Q$ has

[2] Note that the most common case for Q is a formula of first-order real arithmetic, but any dGℒ formula will work. Evolution domain constraints Q turn out to be unnecessary, because they can be defined

no effect on the state, except that Angel loses the game immediately if dGL formula Q does not hold in the current state, because she failed the test she was supposed to pass. The test game $?Q$ challenges Angel and she loses immediately if she fails. Angel does not win just because she passed the challenge $?Q$, but at least the game continues. So passing challenges is a necessary condition to win games. Failing challenges, instead, immediately makes Angel lose.

The *compound games* of dGL are sequential, choice, repetition, and duals. The *sequential game* $\alpha;\beta$ is the hybrid game that first plays hybrid game α and, when hybrid game α terminates without a player having won already (so no challenge in α failed), continues by playing game β. When playing the *choice game* $\alpha \cup \beta$, Angel chooses whether to play hybrid game α or play hybrid game β. Like all the other choices, this choice is dynamic, i.e. every time $\alpha \cup \beta$ is played, Angel gets to choose again whether she wants to play α or β this time. The *repeated game* α^* plays hybrid game α repeatedly and Angel chooses, after each play of α that terminates without a player having won already, whether to play the game again or not, albeit she cannot choose to play indefinitely but has to stop repeating ultimately. Angel is also allowed to stop α^* right away after zero iterations of α. Most importantly, the *dual game* α^d is the same as playing the hybrid game α with the roles of the players swapped. That is Demon decides all choices in α^d that Angel has in α, and Angel decides all choices in α^d that Demon has in α. Players who are supposed to move but deadlock lose. Thus, while the test game $?Q$ causes Angel to lose if formula Q does not hold, the *dual test game* (or *dual challenge*) $(?Q)^d$ instead causes Demon to lose if Q does not hold.

For example, if α describes the game of chess, then α^d is chess where the players switch sides. If α, instead, describes the hybrid game corresponding to your lab 5 robot model where you are controlling a robot and your course instructors are controlling the roguebot, then α^d describes the dual game where you take control of the roguebot and the course instructors are stuck with your robot controls.

The dual operator ^{d} is the only syntactic difference of dGL for hybrid games compared to dL for hybrid systems [Pla08, Pla12a], but a fundamental one [Pla15], because it is the only operator where control passes from Angel to Demon or back. Without ^{d} all choices are resolved uniformly by Angel without interaction. The presence of ^{d} requires a thorough semantic generalization throughout the logic to cope with such flexibility.

5 Differential Game Logic

Hybrid games describe how the world can unfold when Angel and Demon interact according to their respective control choices. They explain the rules of the game how Angel and Demon interact, but not who wins the game, nor what the respective objec-

using hybrid games [Pla15]. In the ordinary differential equation $x' = f(x)$, the term x' denotes the time-derivative of x and $f(x)$ is a polynomial term that is allowed to mention x and other variables. More general forms of differential equations are possible [Pla10a, Pla10b], but will not be considered explicitly.

tives of the players are.[3] The winning conditions are specified by logical formulas of differential game logic. Modal formulas $\langle\alpha\rangle\phi$ and $[\alpha]\phi$ refer to hybrid games and the existence of winning strategies for Angel and Demon, respectively, in a hybrid game α with a winning condition specified by a logical formula ϕ.

Definition 2 (dGℒ formulas)**.** The *formulas of differential game logic* dGℒ are defined by the following grammar (ϕ, ψ are dGℒ formulas, p is a predicate symbol of arity k, θ_i are (polynomial) terms, x a variable, and α is a hybrid game):

$$\phi, \psi \ ::= \ p(\theta_1, \ldots, \theta_k) \mid \theta_1 \geq \theta_2 \mid \neg\phi \mid \phi \wedge \psi \mid \exists x\, \phi \mid \langle\alpha\rangle\phi \mid [\alpha]\phi$$

Other operators $>, =, \leq, <, \vee, \rightarrow, \leftrightarrow, \forall x$ can be defined as usual, e.g., $\forall x\, \phi \equiv \neg\exists x\, \neg\phi$. The modal formula $\langle\alpha\rangle\phi$ expresses that Angel has a winning strategy to achieve ϕ in hybrid game α, i.e. Angel has a strategy to reach any of the states satisfying dGℒ formula ϕ when playing hybrid game α, no matter what strategy Demon chooses. The modal formula $[\alpha]\phi$ expresses that Demon has a winning strategy to achieve ϕ in hybrid game α, i.e. a strategy to reach any of the states satisfying ϕ, no matter what strategy Angel chooses.[4] Note that the same game is played in $[\alpha]\phi$ as in $\langle\alpha\rangle\phi$ with the same choices resolved by the same players. The difference between both dGℒ formulas is the player whose winning strategy they refer to. Both use the set of states where dGℒ formula ϕ is true as the winning states for that player. The winning condition is defined by the modal formula, α only defines the hybrid game form, not when the game is won, which is what ϕ does. Hybrid game α defines the rules of the game, including conditions on state variables that, if violated, cause the present player to lose for violation of the rules of the game. The dGℒ formulas $\langle\alpha\rangle\phi$ and $[\alpha]\neg\phi$ consider complementary winning conditions for Angel and Demon.

6 Demon's Controls

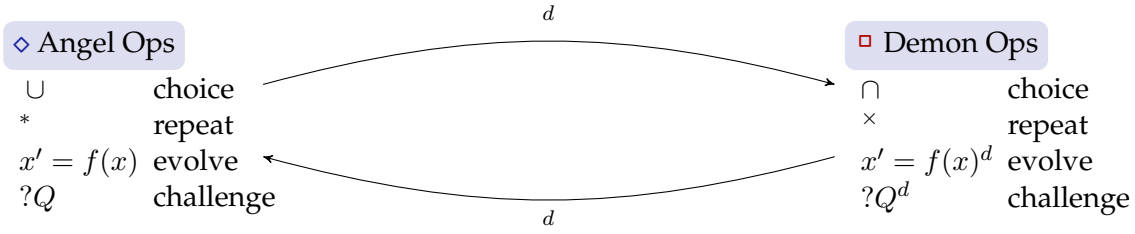
Angel has full control over all choices in each of the operators of hybrid games *except* when the operator ^{d} comes into play. All choices within the scope of (an odd number of) ^{d} belong to Demon, because ^{d} makes the players switch sides. Demon's controls, i.e. direct controls for Demon, can be defined using the duality operator ^{d} on Angel's controls.

Demonic choice between hybrid game α and β is $\alpha \cap \beta$, defined by $(\alpha^d \cup \beta^d)^d$, in which either the hybrid game α or the hybrid game β is played, by Demon's choice. The choice for the $\cup$ operator belongs to Angel, yet since it is nested within ^{d}, that choice goes to Demon, except that the ^{d} operators around α and β restore the original ownership of controls. *Demonic repetition* of hybrid game α is $\alpha^\times$, defined by $((\alpha^d)^*)^d$, in which α is repeated as often as Demon chooses to. Again, the choice in the * operator belongs to

[3] Except that players lose if they disobey the rules of the game by failing their respective challenges.

[4] It is easy to remember which modal operator is which. The formula $\langle\alpha\rangle\phi$ clearly refers to Angel's winning strategies because the diamond operator $\langle\cdot\rangle$ has wings.

Angel, but in a ^{d} context goes to Demon, while the choices in the α, β subgames underneath stay as they were originally thanks to the additional ^{d} operators. In $\alpha^\times$, Demon chooses after each play of α whether to repeat the game, but cannot play indefinitely so he has to stop repeating ultimately. The *dual differential equation* $(x' = \theta \,\&\, Q)^d$ follows the same dynamics as $x' = \theta \,\&\, Q$ except that Demon chooses the duration, so he cannot choose a duration during which Q stops to hold at any time. Hence he loses when Q does not hold in the current state. Dual assignment $(x := \theta)^d$ is equivalent to $x := \theta$, because it never involved any choices to begin with. Angel's control operators and Demon's control operators correspond to each other by duality:



7 Operational Game Semantics (informally)

Treatment of a proper semantics for differential game logic will be deferred to the next lecture. A graphical illustration of the choices when playing hybrid games is depicted in Fig. 1. The nodes where Angel gets to decide are shown as diamonds ◇, the nodes where Demon decides are shown as boxes □. Circle nodes are shown when it depends on the remaining hybrid game which player it is that gets to decide. Dashed edges indicate Angel's actions, solid edges would indicate Demon's actions, while zigzag edges indicate that a hybrid game is played and the respective players move as specified by that game. The actions are the choice of time for $x' = f(x) \,\&\, Q$, the choice of playing the left or the right game for a choice game $\alpha \cup \beta$, and the choice of whether to stop or repeat in a repeated game α^*. This principle can be made rigorous in an operational game semantics [Pla15], which conveys the intuition of interactive game play for hybrid games, relates to game theory and descriptive set theory, but is also beyond the scope of these lecture notes. Observe how all choices involve at most two possibilities except differential equations, which have an uncountably infinite branching factor, one option for each duration $r \in \mathbb{R}$.

As an example, consider the *filibuster formula*:

$$\langle (x := 0 \cap x := 1)^* \rangle x = 0 \tag{1}$$

It is Angel's choice whether to repeat (*), but every time Angel repeats, it is Demon's choice ($\cap$) whether to play $x := 0$ or $x := 1$. What is the truth-value of the dGℒ formula (1)?

The game in this formula never deadlocks, because every player always has a remaining move (here even two). But it may appear as if the game had perpetual checks,

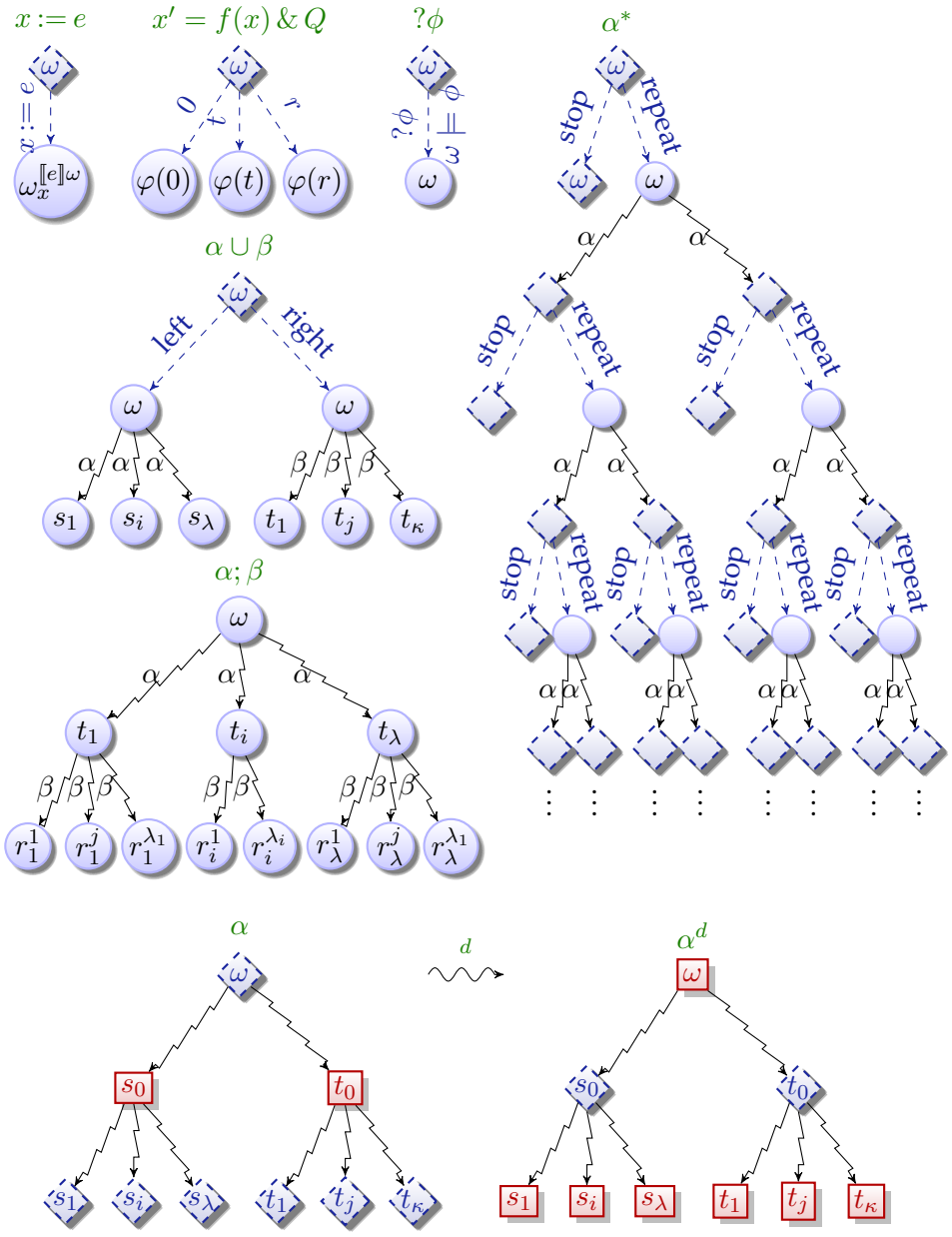


Figure 1: Operational game semantics for hybrid games of dGℒ

because no strategy helps either player win the game; see Fig. 2. How could that happen and what can be done about it?

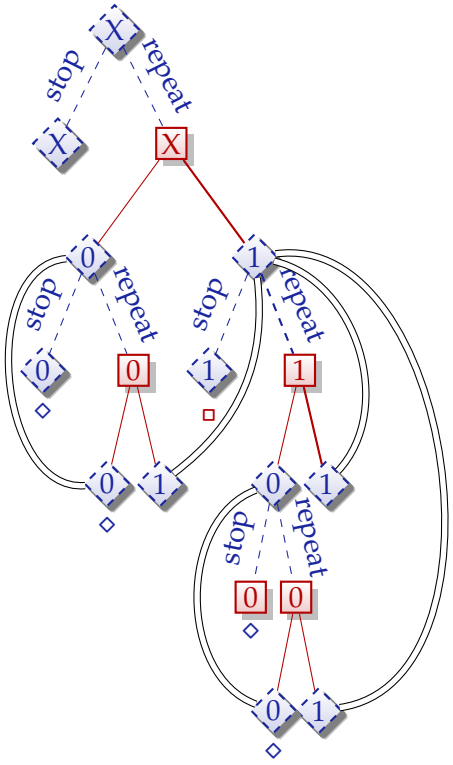


Figure 2: The filibuster game formula $\langle(x := 0 \cap x := 1)^*\rangle x = 0$ looks like it might be non-determined and not have a truth-value (unless $x = 0$ initially) when the strategies follow the thick actions. Angel's action choices are illustrated by dashed edges from dashed diamonds, Demon's action choices by solid edges from solid squares, and double lines indicate identical states with the same continuous state and a subgame of the same structure of subsequent choices. States where Angel wins are marked ◇ and states where Demon wins by ▫.

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

The mystery of the filibuster game can be solved when we remember that the game still ultimately ought to stop in order to be able to inspect who won the game. Angel is in charge of the * repetition and she can decide whether to stop or repeat. The filibuster game has no tests, so the winner only depends on the final state of the game, because both players are allowed to play arbitrarily without having to pass tests in between. Angel wins a game play if $x = 0$ holds in the final state and Demon wins if $x \neq 0$ holds in the final state. What do the strategies indicated in Fig. 2 do? They postpone the end of the game forever, hence there would never be a final state in which it could be evaluated who won. That is, indeed, not a way for anybody to win anything. Yet, Angel was in charge of the repetition *, so it is really her fault if the game never comes to a stop to evaluate who won, because she has to call it quits at some point. Consequently, the semantics of hybrid games requires players to repeat as often as they want but they cannot repeat indefinitely. This will be apparent in the actual semantics of hybrid games, which is defined as a denotational semantics corresponding to winning regions. Thus, (1) is false unless $x = 0$ already holds initially.

The same phenomenon happens in

$$\langle (x := 0; x' = 1^d)^* \rangle x = 0 \tag{2}$$

in which both players can let the other one win. Demon can let Angel win by choosing to evolve for duration 0. And Angel can let Demon win by choosing to stop even if $x \neq 0$. Only because Angel will ultimately have to stop repeating does the formula in (2) have a truth-value and the formula is false unless $x = 0$ already holds initially.

It is of similar importance that the players cannot decide to follow a differential equation forever (duration ∞), because that would make

$$\langle (x' = 1^d; x := 0)^* \rangle x = 0 \tag{3}$$

non-determined. If players were allowed to evolve along a differential equation forever (duration ∞), then Demon would have an incentive to evolve along $x' = 1^d$ forever in the continuous filibuster (3), because as soon as he stops, Angel would would win because of the subsequent $X := 0$. But Angel cannot win without Demon stopping. Since Demon can evolve along $x' = 1^d$ for any finite amount of time he wants, he will ultimately have to stop so that Angel wins and (3) is valid.

8 Summary

This lecture saw the introduction of differential game logic, which extends the familiar differential dynamic logic with capabilities of modeling and understanding hybrid games. Hybrid games combine discrete dynamics, continuous dynamics, and adversarial dynamics. Compared to hybrid systems, the new dynamical aspect of adversarial dynamics is captured entirely by the duality operator ^{d}. Without it, hybrid games are single-player hybrid games, which are equivalent to hybrid systems.

After this lecture showed an informal and intuitive discussion of the actions that hybrid games allow, the next lecture gives a proper semantics to differential game logic and their hybrid games.

Exercises

Exercise 1. Single player hybrid games, i.e. ^{d}-free hybrid games, are just hybrid programs. For each of the following formulas, convince yourself that it has the same meaning, whether you understand it as a differential dynamic logic formula with a hybrid systems or as a differential game logic formula with a hybrid game (that happens to have only a single player):

$$\langle x := 0 \cup x := 1\rangle x = 0$$
$$[x := 0 \cup x := 1]x = 0$$
$$\langle (x := 0 \cup x := 1); ?x = 1\rangle x = 0$$
$$[(x := 0 \cup x := 1); ?x = 1]x = 0$$
$$\langle (x := 0 \cup x := 1); ?x = 0\rangle x = 0$$
$$[(x := 0 \cup x := 1); ?x = 0]x = 0$$
$$\langle (x := 0 \cup x := 1)^*\rangle x = 0$$
$$[(x := 0 \cup x := 1)^*]x = 0$$
$$\langle (x := 0 \cup x := x + 1)^*\rangle x = 0$$
$$[(x := 0 \cup x := x + 1)^*]x = 0$$

Exercise 2. Consider the following dGL formulas and identify under which circumstance they are true?

$$\langle (x := x + 1; (x' = x^2)^d \cup x := x - 1)^*\rangle\, (0 \leq x < 1)$$
$$\langle (x := x + 1; (x' = x^2)^d \cup (x := x - 1 \cap x := x - 2))^*\rangle (0 \leq x < 1)$$
$$\langle (x := x + 1; (x' = x^2)^d \cup (x := x - 1 \cap x := x - 2))^*\rangle (0 < x \leq 1)$$

Exercise 3. The following dGL formula characterizes a one-dimensional game of chase of a robot at position x and a robot at position y, each with instant control of the velocity v among $a, -a, 0$ for x (Angel's choice) and velocity w among $b, -b, 0$ for y (Demon's subsequent choice). The game repeats any number of control rounds following Angel's choice (*). Angel is trying for her robot x to be close to Demon's robot y. Under which circumstance is the formula true?

$$\Big\langle \big(\; (v := a \cup v := -a \cup v := 0);$$
$$\quad (w := b \cap w := -b \cap w := 0);$$
$$\quad x' = v, y' = w \; \big)^* \Big\rangle\, (x - y)^2 \leq 1$$

Exercise 4 (*). The following dGL formula characterizes a two-dimensional game of chase of a robot at position (x_1, x_2) facing in direction (d_1, d_2) and a robot at position (y_1, y_2) facing in direction (e_1, e_2). Angel has direct control over the angular velocity ω among $1, -1, 0$ for robot (x_1, x_2) and, subsequently, Demon has direct control over the angular velocity ϱ among $1, -1, 0$ for robot (y_1, y_2). The game repeats any number of

control rounds following Angel's choice (*). Angel is trying for her robot to be close to Demon's robot. Is the following dGℒ formula valid? Can you identify some circumstances under which it is true? Or some circumstances under which it is false? How does this formula relate to lab 4?

$$\Big\langle \Big((\omega := 1 \cup \omega := -1 \cup \omega := 0);$$
$$(\varrho := 1 \cap \varrho := -1 \cap \varrho := 0);$$
$$(x_1' = d_1, x_2' = d_2, d_1' = -\omega d_2, d_2' = \omega d_1, y_1' = e_1, y_2' = e_2, e_1' = -\varrho e_2, e_2' = \varrho e_1)^d \Big)^* \Big\rangle$$
$$(x_1 - y_1)^2 + (x_2 - y_2)^2 \leq 1$$

References

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

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

[Pla10a] André Platzer. Differential-algebraic dynamic logic for differential-algebraic programs. *J. Log. Comput.*, 20(1):309–352, 2010. doi:10.1093/logcom/exn070.

[Pla10b] André Platzer. *Logical Analysis of Hybrid Systems: Proving Theorems for Complex Dynamics*. Springer, Heidelberg, 2010. doi:10.1007/978-3-642-14509-4.

[Pla11] André Platzer. Stochastic differential dynamic logic for stochastic hybrid programs. In Nikolaj Bjørner and Viorica Sofronie-Stokkermans, editors, *CADE*, volume 6803 of *LNCS*, pages 431–445. Springer, 2011. doi:10.1007/978-3-642-22438-6_34.

[Pla12a] André Platzer. The complete proof theory of hybrid systems. In LICS [LIC12], pages 541–550. doi:10.1109/LICS.2012.64.

[Pla12b] André Platzer. Dynamic logics of dynamical systems. *CoRR*, abs/1205.4788, 2012. arXiv:1205.4788.

[Pla12c] André Platzer. Logics of dynamical systems. In LICS [LIC12], pages 13–24. doi:10.1109/LICS.2012.13.

[Pla15] André Platzer. Differential game logic. *ACM Trans. Comput. Log.*, 17(1):1:1–1:51, 2015. doi:10.1145/2817824.