Recitation 7: Hybrid Games
15-424/15-624/15-824 Logical Foundations of Cyber-Physical Systems

Notes by Brandon Bohrer.
Edits by Yong Kiam Tan, (later) Katherine Cordwell (kcordwel@andrew.cmu.edu), and Aditi Kabra

1 Hybrid Games

Note: We started off this section by watching Geri's Game from Pixar. Watch it online if you have not already done so. It is a useful illustration of how the dual operator works.

The syntax of hybrid games extends that of hybrid programs with the dual operator:

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

At first glance, this is a simple syntactic extension. We have seen (and will review later in these notes), however, that the semantics of hybrid games is radically different from that of hybrid programs. This is a timely reminder that the meaning of a piece of syntax is really only given by its semantics and nothing else. We have not yet looked the semantics so this recitation will mainly focus on modeling modeling some simple scenarios.

First, let us recap the useful Demon versions of operators that we have seen in class:

Angel	**Demon**	**Definition with Duals**
$\alpha \cup \beta$	$\alpha \cap \beta$	$(\alpha^d \cup \beta^d)^d$
α^*	$\alpha^\times$	$((\alpha^d)^*)^d$
$\alpha; \beta$	$\alpha; \beta$	$\alpha; \beta$
$x := e$	$x := e$	$x := e$
$\{x' = f(x) \,\&\, Q\}$	$\{x' = f(x) \,\&\, Q\}^d$	$\{x' = f(x) \,\&\, Q\}^d$
$?H$	$?H^d$	$?H^d$

Notice that the sequential composition and assignment operators do not have duals. Intuitively, there is no choice involved for these operators, so it does not matter who is making the choice.

What about $\alpha \cap \beta \equiv (\alpha^d \cup \beta^d)^d$? Well, that's saying Angel flips the chessboard and her opponent gets to make a choice—so after the flip, Demon sees $\alpha^d \cup \beta^d$. Then he makes his choice and flips the board back to Angel, so then she is left with either $((\alpha)^d)^d$ or $((\beta)^d)^d$; i.e. α or β.

Similarly, for $((\alpha^d)^*)^d$, Angel flips the chessboard. Demon wants to win in $(\alpha^d)^*$. He chooses whether to end the game or play another iteration of α, and then flips the chessboard back to Angel. If Demon decided not to loop, the game ends and Angel have to see if she won in whatever state she's in. If Demon chose to play, then the game continues with Angel playing α.

Here is an example: $(x := 3 \cup x := 4) \cap x := 2)$ is $((x := 3 \cup x := 4)^d \cup (x := 2)^d)^d$. When Angel flips the board, Demon sees $(x := 3 \cup x := 4) \cap x := 2)$ is $((x := 3 \cup x := 4)^d \cup (x := 2)^d)$. He chooses which branch and returns control to Angel. Angel then sees either $((x := 3 \cup x := 4)^d)^d$ or $((x := 2)^d)^d$, i.e., $x := 3 \cup x := 4$ or $x := 2$, and plays accordingly.

2 Winning Hybrid Games

We have also seen in class that $\langle\alpha\rangle P$ means "Angel wins" while $[\alpha]P$ means "Demon wins". Let us think about the dual operators with respect to these modalities, first in a few small exercises and then in a modeling example.

2.1 Exercises

Exercise 1:
When is $\langle ?H^d\rangle P$ true? What about $[?H^d]P$?
Answer: In the former, Angel wins if H is false in the current state, or if P is true (i.e., $H \rightarrow P$). In the latter, Demon wins if the test succeeds and also P is true (i.e., $H \wedge P$).

Exercise 2:
When is $\langle\alpha \cap \beta\rangle P$ true? What about $[\alpha \cap \beta]P$?
Answer: For $\langle\alpha \cap \beta\rangle P$, we require that no matter if Demon picked program α or β, Angel can play that game to win into postcondition P. In contrast, $[\alpha \cap \beta]P$ is true if Demon can pick either α or β so that no matter how the game unfolds in the chosen game, Demon wins into postcondition P.

Exercise 3:
When is $\langle\{x' = f(x) \,\&\, Q\}^d\rangle P$ true? What about $[\{x' = f(x) \,\&\, Q\}^d]P$?
Answer: In the former, Demon controls how long the ODEs are ran. Therefore, Angel wins if for all runs of the ODE the postcondition P is true at the end of the solution. In the latter, Demon can choose how long to run the ODE, and so Demon wins when there is some run of the ODE such that postcondition P is true.[1]
Note: Evolution domain constraints can be expressed by the other operators in dGL (review how!), but they're still useful to have in the syntax.

[1]All of the above can be seen from the following derived axiom of dGL, which we will see later in class. $\langle\alpha^d\rangle P \leftrightarrow [\alpha]P$.

2.2 A Bus Chase

To help motivate dGL and give an example of when we would use it to model, we will be interested in the following real life scenario[2]: Suppose that Brandon lives some distance d away from the school. Each morning, he has two choices for getting to school: 1) take the bus, 2) walk towards school. Of course, if Brandon gets tired along the way, then he could also stop walking and just wait for the bus to arrive.

We would like to know if there a strategy for Brandon so that he can always get to school on time, e.g., within T minutes. Let us model this situation more precisely. We will assume that the current position of Brandon is x (with his house at $x = 0$), the bus is at position b (initially $b < 0$), and the school is at position d. For simplicity, the school is infinitely large so Brandon is in the school whenever $x \geq d$.

We can write a discrete controller for Brandon that models his instantaneous decision to walk or wait for the bus. In addition, if Brandon is already on the bus (which we model by $x = b$), then his velocity is faster (because the bus drives quickly). Brandon's walking speed is given by W, while the bus cruising speed is given by V.

$$\alpha \equiv ?x = b; v_x := V \cup v_x := W \cup v_x := 0$$

Exercise 4:
If Brandon is already on the bus, what do the choices $v_x := W \cup v_x := 0$ mean?
Answer: Intuitively, this just says that Brandon gets off the bus and starts walking (or waits for the next bus). It is not clear why Brandon would want to do this, but it is part of the control allowed in our model.

Next, let us model the bus. This will be a demonic bus, and so it really does not want you to be able to board it. However, if Brandon has already boarded the bus, then it is forced to move at its cruising speed. Otherwise, all bets are off and the bus could accelerate, brake or even reverse arbitrarily as it pleases.

$$\beta \equiv \textbf{if } x = b \textbf{ then } v_b := V; a_b := 0 \textbf{ else } a_b := *$$

Finally, after Brandon and the bus have both made their decision, we will let the physics of the real world run:

$$\gamma \equiv \{x' = v_x, b' = v_b, v_b' = a_b, t' = 1\}$$

We can now put these three programs together in a loop and start our timer off at $t = 0$:

$$\delta \equiv t := 0; (\alpha; \beta; \gamma)^*$$

Exercise 5:

[2]According to a former TA, Brandon Bohrer. The current TA agrees.

How could we express the property that Brandon has a strategy to get to school within T minutes in dL?
Answer: This is a trick question, because we will not be able express this cleanly. However, we shall try to do it anyway.

Recall from Lab 1 that we represented the efficiency property for the controller with a formula of the form $[\alpha](v = 0 \to \cdots)$. We can do something similar here. For example, we could say that if the timer has reached time $t = T$, then Brandon's position must be $x \geq d$. Let Γ_{const} contain all relevant reasonable assumptions on constants (e.g., $d > 0$, etc.). Then we could try to model our situation in the following formula:

$$\Gamma_{\text{const}} \vdash [\delta](t = T \to x \geq d)$$

There are two flaws with this formula. First, the box modality quantifies over *all runs* of the program. This is not quite what we want, because there is a run where Brandon simply stays and waits for the bus forever! This formula is actually unsatisfiable assuming appropriate initial assumptions in Γ_{const}.

Secondly, the postcondition intuitively says **if** $t = T$ then $x \geq d$. This is somewhat unsatisfying (and not best practice), because we do not actually know that $t = T$ is eventually reached. Admittedly, this is rather obvious for this model because if the loop (and ODE) is executed for sufficiently many iterations we should be able to reach $t = T$. This would not be the case for more complicated models, though.

To resolve both of these issues, we could try to instead ask a diamond modality formula:

$$\Gamma_{\text{const}} \vdash \langle\delta\rangle(t = T \land x \geq d)$$

Exercise 6:
What is wrong with this formula?
Answer: Just like the box modality quantifies over *all runs*, the diamond modality only quantifies over *some* run of δ. In particular, we have lost the adversarial dynamics of the demonic bus! We could instead try to break the hybrid program and write it down with alternating modalities:

$$\Gamma_{\text{const}} \vdash \langle\alpha\rangle[\beta]\langle\gamma\rangle(t = T \land x \geq d)$$

These alternations correctly capture what we want: there is *some* run of Brandon's controller, such that *for all* runs of the bus controller, there is *some* run of the physics so that $t = T \land x \geq d$. But we have dropped the program loop! Now all we can do is run the controllers and ODEs once, which is not very interesting. Intuitively, we want them to alternate as many times as required until the timer expires.

To correctly express our intended meaning for the model, we will therefore need to use a hybrid game rather than just a simple hybrid program. In particular, let us think about the formula:

$$\langle t := 0; (\alpha; \beta^d; \gamma)^*\rangle(t = T \land x \geq d)$$

This says that the Angel player (in this case, Brandon) has a strategy to win into post-condition $t = T \land x \geq d$ in the hybrid game $(\alpha; \beta^d; \gamma)^*$.

Let us peel apart the game layer by layer and read it intuitively in the context of what we are trying to model.

First, the outermost loop operator gives Brandon a choice of repeating the loop as many times as possible. This seems natural, because after all, it is Brandon who is heading to school.

Next, in the body of the loop, Brandon first gets to run his controller α. Control is then ceded over to the bus controller by use of the dual operator. Brandon has no control over what the demonic bus does, so in order for Brandon to win, he must be able to cope with whatever strategy the bus decides to play. After the bus controller has made its choice, control is returned to Brandon who runs the physics.

Exercise 7:
That latter step where Brandon controlled physics seems rather fishy. Why? Should we add the dual operator on γ as well?
Answer: Adding a dual operator here would not quite work because then the demonic bus can always decide to run the ODEs for no time, causing Brandon to lose the game. As we saw in class, this is a filibuster strategy by the Demon that prevents Brandon from ever achieving his objecting. In this case, our model is simply saying that Brandon can make his choice of whether to stop or walk at any time he sees fit.

3 Hybrid Game Semantics

The semantics of a formula $[\![P]\!]$ is (as usual) the set of states in which that formula is true. The key difference when working with hybrid games is in the semantics of the modal connectives. They now make use of the winning regions $\varsigma_\alpha, \delta_\alpha$:

$$[\![\langle\alpha\rangle P]\!] = \varsigma_\alpha([\![P]\!])$$
$$[\![[\alpha]P]\!] = \delta_\alpha([\![P]\!])$$

Exercise 8:
For revision purposes, it could help to fill in the following table which compares the transition semantics $[\![\alpha]\!]$ against the winning region semantics. Since there are many entries to fill, informal definitions will suffice, but make sure you know how to formally write down each of the definitions. We use X to denote an arbitrary set of states.

α	$[\![\alpha]\!]$	$\varsigma_\alpha(X)$	$\delta_\alpha(X)$
$x := e$			
$?Q$			
$\{x' = f(x) \,\&\, Q\}$			
$\alpha \cup \beta$			
$\alpha;\beta$			
α^*			
α^d	N/A		

Answer: The transition semantics should be clear by now, so we will focus on the winning regions, and especially the winning regions for Demon. The main intuition to keep in mind that these winning regions are defined from the point of view of Angel playing the game.
Note: As α^d doesn't have a corresponding hybrid program, it doesn't have a transition semantics, so the bottom left slot of the table should be empty.

Recall that $\varsigma_\alpha(X), \delta_\alpha(X)$ are the sets of states for which Angel and Demon can win into X by playing game α respectively. The target region is the set X in all of the following answers.

- ($x := e$) **Question:** Is the following correct? Angel wins if from initial state ω the assignment leads to a winning state, i.e., $\omega_x^{\omega[\![e]\!]} \in X$. Conversely, Angel loses, and thus Demon wins if $\omega_x^{\omega[\![e]\!]} \notin X$.

  Answer: **That is incorrect!** Remember that both Angel and Demon are trying to win into X, and the assignment game has no choices for either of them to make. Thus, Demon actually wins into X when $\omega_x^{\omega[\![e]\!]} \in X$ as well. Formally:

$$\varsigma_{x:=e}(X) = \delta_{x:=e}(X) = \{\omega \mid \omega_x^{\omega[\![e]\!]} \in X\}$$

- ($?Q$) Angel wins when both Q and X are true in the initial state because Angel must pass the test. Conversely, Demon wins when either Q is false, or X is true in the initial state. Formally:

$$\varsigma_{?Q}(X) = [\![Q]\!] \cap X$$
$$\delta_{?Q}(X) = [\![Q]\!]^\complement \cup X$$

- ($\{x' = f(x) \,\&\, Q\}$) Angel wins if there is **some** solution to the ODE that stays in the domain constraint Q for its entire duration, reaching the target region X at its endpoint. Conversely, Demon wins if **all** such solutions satisfy X at their endpoints. (Formal version omitted)

- ($\alpha \cup \beta$) Recall that Angel gets to make the choices, so Angel wins if she can win into X by choosing either to play the α or β game. Conversely, Demon needs to win into X regardless of which game Angel chooses to play.

$$\varsigma_{\alpha\cup\beta}(X) = \varsigma_\alpha(X) \cup \varsigma_\beta(X)$$
$$\delta_{\alpha\cup\beta}(X) = \delta_\alpha(X) \cap \delta_\beta(X)$$

- $(\alpha;\beta)$ Like assignments, neither player has much choice in the sequential game. Thus, the definitions are straightforward compositions of winning regions.

  $$\varsigma_{\alpha;\beta}(X) = \varsigma_\alpha(\varsigma_\beta(X))$$
  $$\delta_{\alpha;\beta}(X) = \delta_\alpha(\delta_\beta(X))$$

- (α^*) Loops are easily the most complicated part of the semantics of hybrid games. If you have never seen least or greatest fixpoints before, it is perhaps easiest to remember them via their defining equations.

  For Angel, her winning region is characterized by the **least** set of states containing X and closed under taking "one more iteration" of the loop.

  $$\varsigma_{\alpha^*}(X) = \bigcap_Z (X \cup \varsigma_\alpha(Z) \subseteq Z)$$

  Let us break down this definition:

  1. $X \subseteq Z$ requires the target region X to be contained in Z, since Angel can always win from those states by running the loop 0 times.
  2. $\varsigma_\alpha(Z) \subseteq Z$ intuitively says Z should also be closed under taking "one more iteration" of α. If Angel is at state q and she can win with one more iteration (i.e., $q \in \varsigma_\alpha(X)$), then q should be in the set of all states she can win from.
  3. Finally, the $\bigcap_Z$ operator ensures that $\varsigma_{\alpha^*}(X)$ is the smallest set Z satisfying the previous two properties. This is well-defined because if two sets U and V satisfy 1) and 2), then $U \cap V$ also satisfies 1) and 2).

  However, these three points don't make much sense in isolation—they have to be considered together.

Some further discussion

Why do we want the smallest set Z? First, notice that the set of all states satisfies both 1) and 2), but it certainly doesn't capture information about α—so it's not the right semantics. The fact that we take the smallest set captures that Angel can't win the game by filibustering.

But why the smallest and not the second smallest? Say A is the smallest set that satisfies 1) and 2), and say that B is a slightly larger set that satisfies 1) and 2). Then take a state $y \in B$ that's not in A.

Using property 2), $y \notin \varsigma_\alpha(A)$. But all winning states are contained in A. So that means you can't win from y with 1 repetition of α. But also from 2), the state we get to from y after one repetition of α is in B. Then, by the same argument we just saw, you can't win from y with 2 repetitions either. And so on. So this y shouldn't be in our loop semantics.

Note: This is NOT a formal proof; it's just some intuition. I just want to convince you that the least fixpoint is the right semantics for$\varsigma_\alpha(X)$. You should think about the definition and the semantics and form your own intuition!

Conversely, Demon must be able to win regardless of how many times Angel chooses to run the loop. Thus, the winning region is characterized by a **greatest** set of states where Demon wins even if Angel decides to run the loop for "one more iteration".

$$\delta_{\alpha^*}(X) = \bigcup_Z (Z \subseteq X \cap \delta_\alpha(Z))$$

Let's break down this definition:

1. $Z \subseteq X$ requires that Z is in the target region X already, because otherwise Angel can make Demon lose by running the loop 0 times.
2. $Z \subseteq \delta_\alpha(Z)$ can be read as follows. Suppose Angel runs one more iteration of the loop from Z, then consider Demon's winning region $\delta_\alpha(Z)$. It must be the case that Z is already contained in $\delta_\alpha(Z)$ as well, because Angel could otherwise make Demon lose with one more iteration.
3. Finally, the $\bigcup_Z$ operator ensures that $\delta_{\alpha^*}(X)$ is the largest set Z satisfying the previous two properties. Why do we want this? Well, in each Z that satisfies 1) and 2), Angel can't win now or in the future. So we take the union of all such Z.

Some further discussion

Suppose that we had a state ω with the special property that playing the game α one more time always leads us back to ω (regardless of the moves either player makes).

For Angel, suppose that $\omega \notin X$, so clearly Angel cannot win into X no matter how many iterations she wants to run the loop for. Now, consider a pre-fixpoint set of states Z with $X \cup \varsigma_\alpha(Z) \subseteq Z$ and consider $Y = Z \cup \{\omega\}$, then Y is also a pre-fixpoint but it is clearly not what the winning region for Angel should be because ω is included in Y. The $\bigcap$, or least fixpoint, operation ensures that none of these extra "junk" states are included.

For Demon, suppose that $\omega \in X$, so Demon wins into X no matter how many iteration Angel runs the loop for. Now, consider a post-fixpoint set of states Z with $X \cup \varsigma_\alpha(Z) \subseteq Z$ and consider $Y = Z \setminus \{\omega\}$, then Y is also a post-fixpoint, but it is now missing the state ω from which Demon certainly wins from. The $\bigcup$, or greatest fixpoint, operation ensures that all of these "consistent" states are included.

Note: Again, this is NOT a formal proof; it's just some intuition. You should think about the definition and the semantics and form your own intuition!

- (α^d) The crucial ingredient that makes hybrid games work so elegantly is the dual operator. This operator has no hybrid program counterpart so its corresponding transition semantics in the table you filled in above should be empty.

  The idea behind its semantics is illustrated in Geri's Game which we watched in the last recitation. For the dual game α^d, Angel flips the board around (taking on the role of Demon) and tries to win the game α into $X^\complement$ instead. If the demonic Angel wins game α into $X^\complement$, then the regular Angel loses that game. Conversely, if the demonic Angel has no way to win the game α into $X^\complement$, then regular Angel wins. Thus:

$$\varsigma_{\alpha^d}(X) = (\varsigma_\alpha(X^\complement))^\complement$$

  Similarly for Demon (think through the intuition from Geri's Game yourselves!):

$$\delta_{\alpha^d}(X) = (\delta_\alpha(X^\complement))^\complement$$

The very last case for α^d highlights an important property of the winning region semantics. The "demonic Angel" is really just Demon so we should just think of the Angel changing into the Demon directly.

Formally, we start by showing that hybrid games are consistent and determined:

$$(\varsigma_\alpha(X^\complement))^\complement = \delta_\alpha(X)$$

Intuitively, this says that the set of states where Angel does not have a strategy to win into the set $X^\complement$ by playing game α is exactly those where Demon has a strategy to win into X.

This leads us on to the axiomatics where this is rendered as the determinacy axiom of dGL:

$$([\cdot]) \quad [\alpha]P \leftrightarrow \neg\langle\alpha\rangle\neg P$$

Taken together with the duality axiom:

$$(\langle^d\rangle) \quad \langle\alpha^d\rangle P \leftrightarrow \neg\langle\alpha\rangle\neg P$$

We obtain the following very useful axiom that allows us to switch from proving diamond properties to box properties (and vice-versa) when we encounter the dual operator:

$$\langle\alpha^d\rangle P \leftrightarrow [\alpha]P$$

Note: We spent some time in class going through examples of this box-to-diamond property. You should try out some examples yourself to become convinced that $\langle\alpha^d\rangle P$ is equivalent to $[\alpha]P$ but different from $[\alpha^d]P$.

4 Axiomatics and Proving Hybrid Games

Notice that the $[\cdot]$ determinacy axiom is actually true in dL as well, when α is a hybrid program. In fact, as we have already seen in class, many of the axioms and proof rules of dGL are identical to those of dL, just with a different underlying semantics. This is not always the case: there **are** axioms of dL that would be unsound for hybrid games. However, if the hybrid game α does not mention the duality operator then it is indeed the case that all of the axioms of dL apply.

For now, let us work through some dGL proofs using the axioms that we are already familiar with (avoiding unsound uses of axioms).

4.1 Verified Filibustering

Our first example is a simple game:

$$x = 0 \rightarrow \langle\big((x := 1 \cup v := 1); \{x' = v\}\big)^\times\rangle x = 0$$

Exercise 9:
Is this formula valid? If so, what is Angel's winning strategy?
Answer: It is valid because Angel can always force the value of x to stay at 0 as long as it starts at 0. In other words, Angel can filibuster Demon (who controls the loop). Since Demon must stop running the loop eventually, Demon loses the game.

Now, let us try to verify this formula using the axioms that we already know.

$$x = 0 \vdash \langle\big((x := 1 \cup v := 1); \{x' = v\}\big)^\times\rangle x = 0$$

Exercise 10:
What is the first step?
Answer: Remember that the demonic loop operator gives Demon control over the loop iterations. Thus, the first step we need to use the axioms $[\cdot]$,$\langle^d\rangle$ to remove the outermost duality operator.

$$\dfrac{x = 0 \vdash [\big((x := 1 \cup v := 1); \{x' = v\}\big)^*]x = 0}{x = 0 \vdash \langle\big((x := 1 \cup v := 1); \{x' = v\}\big)^\times\rangle x = 0}\;{\scriptstyle [\cdot],\langle^d\rangle}$$

Note: Hopefully you should have noticed that I have deliberately made a mistake in the proof above by unfolding the demonic loop incorrectly. However, let us see where the proof takes us.

Now that we are left with a loop, all we need to do is to apply loop induction rule, or more formally the ind rule for hybrid games:

$$(\text{ind})\quad \frac{P \to [\alpha]P}{P \to [\alpha^*]P}$$

Exercise 11:
What loop invariant should we use?
Answer: Let us perhaps try the invariant $x = 0$. After all, we do not know much more than that in the beginning. Continuing the proof as we are used to from dL, we can now unfold the game operators:

$$\dfrac{\dfrac{x = 1 \vdash [\{x' = v\}]x = 0 \qquad x = 0 \vdash [\{x' = 1\}]x = 0}{x = 0 \vdash [(x := 1 \cup v := 1); \{x' = v\}]x = 0}\;{\scriptstyle [;],[\cup],[:=],\wedge R}}{x = 0 \vdash [\big((x := 1 \cup v := 1); \{x' = v\}\big)^*]x = 0}\;{\scriptstyle \text{ind}}$$

Exercise 12:
Those premises do not look provable (or even true at all). What went wrong?
Answer: The problem lies with how we unfolded the demonic loop operator. Recall that:

$$\alpha^\times \overset{\text{def}}{\equiv} ((\alpha^d)^*)^d$$

We forgot to include the duality operator around α when unfolding the demonic loop! Always remember to unfold the demonic operators correctly. When in doubt, just remember that the demonic operators are syntactic abbreviations so you could also simply remove all of them before you start your proof.

Here is the corrected unfolding:

$$\dfrac{x = 0 \vdash [\Big(\big((x := 1 \cup v := 1); \{x' = v\}\big)^d\Big)^*]x = 0}{x = 0 \vdash \langle\big((x := 1 \cup v := 1); \{x' = v\}\big)^\times\rangle x = 0}\;{\scriptstyle [\cdot],\langle^d\rangle}$$

Exercise 13:
What is the loop invariant to use?

Answer: As before, $x = 0$ looks like a plausible option. After applying ind we now flip back into a diamond question because of the nested duality operator.

$$\dfrac{\dfrac{x = 0 \vdash \langle (x := 1 \cup v := 1); \{x' = v\}\rangle x = 0}{x = 0 \vdash [\big((x := 1 \cup v := 1); \{x' = v\}\big)^d] x = 0}{\scriptstyle [\cdot],\langle^d\rangle}}{x = 0 \vdash [\Big(\big((x := 1 \cup v := 1); \{x' = v\}\big)^d\Big)^*] x = 0}{\scriptstyle \text{ind}}$$

Now, we can apply the usual axioms you know from dL in diamond form. Since we have not had much experience with the diamond version of these rules, let us do the proof a little slower:[3]

$$\dfrac{\dfrac{x = 0 \vdash \langle x := 1\rangle\langle\{x' = v\}\rangle x = 0, \langle v := 1\rangle\langle\{x' = v\}\rangle x = 0}{x = 0 \vdash \langle (x := 1 \cup v := 1)\rangle\langle\{x' = v\}\rangle x = 0}{\scriptstyle \langle\cup\rangle,\vee R}}{x = 0 \vdash \langle (x := 1 \cup v := 1); \{x' = v\}\rangle x = 0}{\scriptstyle \langle;\rangle}$$

Exercise 14:
What do we do next?
Answer: The key here is to look back at Angel's strategy. If she chooses the left choice, there is not much hope of guaranteeing that $x = 0$ at the end of her run of the ODE unless v is negative (but we do not know that). However, if she chooses the right branch, then since $x = 0$ initially, she can simply evolve the ODE for no time. Thus, we can safely drop the first succedent on the right and complete the proof from the right disjunct using the ODE solution axiom:

$$\dfrac{\dfrac{\dfrac{*}{x = 0 \vdash \exists t \geq 0\, x + t = 0}{\scriptstyle \mathbb{R}}}{x = 0 \vdash \langle\{x' = 1\}\rangle x = 0}{\scriptstyle \langle'\rangle}}{x = 0 \vdash \langle v := 1\rangle\langle\{x' = v\}\rangle x = 0}{\scriptstyle \langle:=\rangle}$$

We used two insights in this proof. First, the duality operators allowed us to flip back and forth between the box and diamond modalities. Second, the proof actually follows Angel's strategy quite closely. Thus, one very good way of figuring out a games proof is to first figure out the Angel (or Demon's) winning strategy.

4.2 Verified Planar Avoidance

Finally, let us turn to modeling situation using hybrid games. The scenario we will be considering is as follows:

The Demon is a mosquito flying around in straight lines and you the human (a.k.a., the Angel) are trying to avoid getting bitten. We want the system to be interactive, so the mosquito could choose to change directions whenever it likes, but whenever it does so, you can also react by changing your direction.
Note: For now, we will skip any bounds on the velocities and position, etc., but you should feel free to extend this model.

[3] Have you seen $\langle;\rangle$ somewhere before?

Exercise 15:
How would we go about modeling this scenario?
Answer: We could start by deciding on the coordinate system. Let us suppose that the coordinates of the mosquito are (x, y) while your coordinates are (a, b), both in the Euclidean plane.

Let us further suppose that both players can instantaneously change their heading direction and velocities, which we will prescribe using v_x, v_y and v_a, v_b respectively.

Exercise 16:
How would we describe the game?
Answer: First, we need to write down controllers for each player. This is relatively simple, since both players simply make a choice of heading.

$$\alpha_m \equiv v_x := *; v_y := *$$

$$\alpha_h \equiv v_a := *; v_b := *$$

Note: This is specific to hybrid games! If a player sees $var := *$, then he/she gets to choose the value for *.

Next, the ODEs describing the motion of the mosquito and the human:

$$\beta \equiv \{x' = v_x, y' = v_y, a' = v_a, b' = v_b\}$$

Exercise 17:
How should we put these three parts of the program together?
Answer: We need to be careful to put the things we want under appropriate player controls. As we said earlier, this mosquito can change its heading anytime it likes, so perhaps α_m, β should both be under its control.

However, we can really only react to the mosquito if we have access to where it is currently heading. So we certainly want α_h to come after α_m.

Finally, we want to avoid getting stung no matter how many times the mosquito changes its heading, so the loop should also be under Demonic control. Thus, our overall program is:

$$\gamma \equiv (\alpha_m^d; \alpha_h; \beta)^\times$$

Exercise 18:
Finally, we are ready to write down (and prove) a formula that means we never get bitten. What should it be?
Answer: There are many possible ways of writing this down. For example, let us say that the human and mosquito have to always be separated by some distance d. Then the dGL formula modeling our scenario could be:

$$d > 0 \land (x-a)^2 + (y-b)^2 > d \to \langle\gamma\rangle (x-a)^2 + (y-b)^2 > d$$

Exercise 19:
Would this formula be valid? If so, what is Angel's strategy?
Answer: There are multiple possibilities for Angel since she has full access over the mosquito's movements. For example, she can simply mirror what the mosquito is doing. In 2-dimensions, she can even do more interesting maneuvers, such as moving perpendicularly away from the mosquito.

Exercise 20:
Add more restrictions to Angel, e.g., bounding her velocity so that simply mirroring the mosquito would not be a viable strategy. Next, prove your resulting model (you can use KeYmaera X to check it).

5 Another Commute Example

Unlike Brandon, when running late, Aditi often uses a segway. Let's explore how we could model the game where Aditi tries to make the segway tip over, but the motor in the segway tries to keep the rigid system consisting of Aditi and the segway upright. We want to check whether given the right restrictions on Aditi's actions, there is always a way for the controller to prevent the system from toppling.

$$\alpha \equiv a = *; ?a \leq a_{max} \land a \geq 0$$

We let Aditi determine an angular acceleration that she exerts on the system, bound above by some constant a_{max}. For simplicity, we are modeling only the situation where she exerts this acceleration in one direction.

$$\beta \equiv F = *$$

The segway motor makes the wheel turn and exert some force F backwards on the floor. The floor resists with some friction F, that pushes the segway forward at the point of contact between the floor and the wheel.

$$\gamma \equiv t := 0; \{\theta' = \omega, \omega' = -(F\cos\theta - N\sin\theta).R - L[mg\sin\theta] - a, t' = 1 \land t \leq T\}$$

We have chosen the axis of rotation to be the axle or the wheel, R to be the radius of the wheel, and L to be the distance between the center of gravity of the system and the axle of the wheel. N is the normal force exerted by the ground upwards, and θ is the angle that the system makes with the axis running perpendicular to the ground. The mass of the entire system is m. We model the system as time triggered; the segway gets a chance to revise its force F every T time at most.

$$P \equiv \theta < \theta_{max} \wedge \theta > \theta_{min}$$

The safety condition is that the system must not tip over too far in either direction.

$$\Gamma \models \langle (\beta; \alpha^d; \delta^d)^\times \rangle P$$

The segway chooses a force to exert, then Aditi decides how to influence the system given the restrictions on angular acceleration that she can provide. She is adversarially trying to make the system topple. Finally, the differential equation evolution happens, and is controlled by Aditi. But she is forced to break out of continuous evolution at least every T time. This loop repeats as many times as Aditi would like. Ultimately, the formula is true if there exists a way for the segway to prevent the angle from getting too large in magnitude.

We discussed some properties that Γ would need to have in oder to make the formula true, though we did not derive a formula for it or attempt a proof.

6 Modeling Prisoner's Dilemma

We modeled prisoner's dilemma, a discrete game, to see that a prisoner doesn't have a "winning" strategy.

$$\alpha \equiv a_d := C \cup a_d := D$$

$$\beta \equiv b_d := C \cup b_d := D$$

Both players have a choice to either cooperate or defect.

$$\begin{aligned}\gamma \equiv & \textbf{if } a_d = C \wedge b_d = C \textbf{ then } j_a = 1 \wedge j_b = 1 \\ & \textbf{else if } a_d = C \wedge b_d = D \textbf{ then } j_a = 3 \wedge j_b = 0 \\ & \textbf{else if } a_d = D \wedge b_d = C \textbf{ then } j_a = 0 \wedge j_b = 3 \\ & \textbf{else if } a_d = D \wedge b_d = D \textbf{ then } j_a = 2 \wedge j_b = 2\end{aligned}$$

γ assigns jail time to each prisoner based on their decisions.

$$P \equiv j_a \leq 1$$

The prisoner A "wins" if they can get off with no more than an year of jail time.

$$\neg\big(\langle \alpha; \beta^d; \gamma \rangle P\big)$$

Prisoner A has no winning strategy.