15-424/15-624 Recitation 1
Logic, Syntax, Semantics and Hybrid Programs

1. **Empty sets!**

   You already know how to work with conjunction and disjunction, i.e. $\phi_1 \wedge \phi_2$ and $\phi_1 \vee \phi_2$. You can keep making the formula bigger by adding conjuncts, such as $\phi_1 \vee \phi_2 \vee \phi_3$, but that can become pretty cumbersome.

   In fact, sometimes it's easier to make a conjunction of all the elements in some set A. If we know that $A = \{\phi_1, \phi_2, \phi_3\}$, then we could rewrite the above example simply as $\bigwedge A$. So much prettier!

   We can now look at what happens when we start removing elements from the set A.

   (a) $\bigwedge A = \phi_1 \wedge \phi_2 \wedge \phi_3$ $\qquad A = \{\phi_1, \phi_2, \phi_3\}$

   (b) $\bigwedge A = \phi_1 \wedge \phi_2$ $\qquad A = \{\phi_1, \phi_2\}$

   (c) $\bigwedge A = \phi_1$ $\qquad A = \{\phi_1\}$

   (d) $\bigwedge A =$??? $\qquad A = \{\}$

   What *do* we do with the empty set? Well, we know that for any conjunction to be true, *all* of its conjuncts need to be true. When $A = \{\}$, are all the conjuncts true? Why, I'm glad you asked! Yes, yes they are, because there aren't any.

   $$\bigwedge \{\} = \text{true}$$

   How about $\bigvee A$ when $A = \{\}$? We can apply a similar reasoning. For a disjunction to be true, at least one of the disjuncts needs to be true. Can we find a disjunct in $A = \{\}$ that is true? Can we find a disjunct *at all*? Nope! So, in fact, it's impossible to satisfy a disjunction under these conditions.

   $$\bigvee \{\} = \text{false}$$

2. **Syntax vs. Semantics. *FIGHT!***

   The difference between syntax and semantics is extremely important.

   Recall that *syntax* refers to what we can write down, the *form*, but not the *meaning*. It's semantics that gives syntax its meaning. The meaning of different things is... well, different! In particular:

   - *Terms*, like $x^3 + xy$, return *values*.
   - *Programs*, like $x := 6; y' = 2 \;\&\; H$, *change state*.
   - *Formulas*, like $\phi \rightarrow (\psi_1 \wedge \psi_2)$, are either *true or false*.

But why is the difference between syntax and semantics important? Because we want to be able to reason and express things about the real numbers $\mathbb{R}$ such as continuous time, or a position in space (this is semantics). And yet... since we are using computers, we are only able to write down things which are finitely representable (this is syntax). Even if you had too much time on your hands and wrote down a lot of digits[1], you still wouldn't be *exact*, and exactness is important for safety!

With syntax we could decide to write anything - any symbols we want! We could even write things like:

$$a \heartsuit b$$

That's syntax. But then we'd have to find a *meaning* for it (semantics), and it turns out that the meaning of love is even trickier than the meaning of cyber-physical systems!

So we decide to stick with some basic arithmetic. Given some state ν, which assigns values to variables, we can get the meaning of addition:

$$[\![\theta_1 + \theta_2]\!]_\nu = [\![\theta_1]\!]_\nu + [\![\theta_2]\!]_\nu$$

Is the $+$ on the left the same as the one on the right? It's confusing, but it isn't. The left one is the *symbol* for addition (syntax), whereas the one on the right is actual addition (semantics). We could've used different symbols, such as $\theta_1 \oplus \theta_2$:

$$[\![\theta_1 \oplus \theta_2]\!]_\nu = [\![\theta_1]\!]_\nu + [\![\theta_2]\!]_\nu$$

This would make it more obvious what is syntax and semantics, but it would also require lots of new symbols. Since we are pros at distinguishing syntax and semantics now, we'll stick with reusing the same symbol while being aware of the difference.

For the formula $\forall x.x > y$, we mean to say that for all possible $d \in \mathbb{R}$, $d > y$ is true. So, could we write the semantics to be the following?

$$\nu \models \forall x.x > y \text{ iff } \nu \models d > y \text{ for all } d \in \mathbb{R}$$

There's a bit of a problem... We just substituted a real number (semantics) into the formula (syntax). That means that because π is a real number, we might have to write it, and we've already established that an eternity of typing is way too much for us.

So, because cyber-physical systems are all about state change, we can try doing that instead.

$$\nu \models \forall x.x > y \text{ iff } \nu[x \mapsto d] \models x > y \text{ for all } d \in \mathbb{R}$$

That's much better! Because state is a semantic entity, we can change that without having to worry too much, and the formula remains remains syntactically acceptable!

[1] `http://www.geom.uiuc.edu/~huberty/math5337/groupe/digits.html`

3. **Differential equations as a program**

Let's try to get an intuitive understanding of the transition relation for ODEs as hybrid programs:

$$(\nu, \omega) \in \rho\left(x' = f\left(x\right) \ \& \ H\right)$$

So, when is it that we can get from ν to ω by following the differential equation with domains specified by $x' = f\left(x\right) \ \& \ H$?

Notice how we know the initial state, ν, making this an initial value problem. Because theorems, we also know that there is a solution to the differential equation, which we will call φ. But what does φ look like? It's a function that will tells us what the state is at each moment of time.

$$\varphi : \mathbb{R}_{\geq 0} \to \mathbb{R}^n$$

Since that's the case, if the differential equation doesn't evolve, then clearly we must be at the initial state!

$$\varphi(0) = \nu$$

Since we are relating ν and ω, then we must somehow be able to reach ω. That means that at some point (in time!), the state must be ω! So, there is some time $t \in \mathbb{R}_{\geq 0}$, such that

$$\varphi(t) = \omega$$

We aren't done yet though, since we haven't handled the domain H. The idea here is that if φ ever leads us to a state outside of H, it's game over. We're not allowed out of H, ever. Didn't you see the stop sign? Geez.

So we must always be within H. Remember that it's φ that tells us "where we are", and that "always" refers to all the time between being at ν and being at ω. In more mathy parlance, what we are saying is that for every $r \in \mathbb{R}$ such that $0 \leq r \leq t$,

$$\varphi(r) \models H$$

These four conditions should hopefully give you an intuitive understanding of how differential equations with domains behave as hybrid programs!