Lecture 2: Syntax

January 24, 2018

We now review the basic definitions of first-order logic in more detail. Recall that a *language* consists of a collection of symbols $\{P_i\}$, each of which has some specified *arity* n_i (which is a nonnegative integer).

Fix, once and for all, an infinite set V of *variables*. Given subset $V_0 \subseteq V$, a *formula of L with free variables in V_0* consists of one of the following five things:

(1) An expression $x = y$, for some pair of elements $x, y \in V_0$ (which might be the same).

(2) An expression $P_i(x_1, \ldots, x_{n_i})$, where P_i is a predicate of the language L having arity n_i, and each x_i is an element of V_0 (the x_i need not be distinct).

(3) An expression $\varphi \vee \varphi'$, where φ and φ' are previously constructed formulae with free variables in V_0.

(4) An expression $\neg\varphi$, where φ is a previously constructed formula with free variables in V_0.

(5) An expression $(\exists x)[\varphi]$, where $x \in V \backslash V_0$, where φ is a previously constructed formula with free variables in $V_0 \cup \{x\}$.

Definition 1. Let L be a language. An *L-structure* is a set M together with a subset $M[P_i] \subseteq M^{n_i}$ for each predicate symbol P_i of L.

Let M be an L-structure, let φ be a formula of L with free variables $\{x_1, \ldots, x_n\}$, and suppose we're given an n-tuple of elements $\vec{c} = (c_1, \ldots, c_n)$ in M. The relation $M \vDash \varphi(\vec{c})$ is defined by recursion as follows:

(1) If φ is the formula $x_i = x_j$, then $M \vDash \varphi(\vec{c})$ if and only if $c_i = c_j$.

(2) If φ is the formula $P_i(x_{j_1}, \ldots, x_{j_{n_i}})$, then $M \vDash \varphi(\vec{c})$ if and only if and only if the tuple $(x_{j_1}, \ldots, x_{j_{n_j}})$ belongs to $M[P_i]$.

(3) $M \vDash (\varphi \vee \varphi')(\vec{c})$ if and only if either $M \vDash \varphi(\vec{c})$ or $M \vDash \varphi'(\vec{c})$.

(4) $M \vDash (\neg\varphi)(\vec{c})$ if and only if it is not true that $M \vDash \varphi(\vec{c})$.

(5) If φ has the form $(\exists y)\psi$ for some formula ψ with free variable $\{x_1, \ldots, x_n, y\}$, the $M \vDash \varphi(\vec{c})$ if and only if there exists some element $a \in M$ such that $M \vDash \psi(\vec{c}, a)$.

We let $M[\varphi]$ denote the set $\{(c_1, \ldots, c_n) \in M^n : M \vDash \varphi(c_1, \ldots, c_n)\}$.

Definition 2. A *sentence* in a language L is a formula φ with no free variables.

A *theory* T consists of a language L, together with a collection of sentences in the language L (which we refer to as the *axioms of* T.

A *model* of a theory T is an L-structure M such that $M \vDash \varphi$ for each axiom φ of T. In this case, we write $M \vDash T$.

Notation 3 (Substitutions of Variables)**.** We will typically denote a formula by $\varphi(\vec{x})$, where $\vec{x} = (x_1, \ldots, x_n)$ is an enumeration of the set of free variables in φ. However, we will constantly engage in a certain abuse of this notation. Suppose that $\varphi(\vec{x})$ is a formula with free variables in a set of variables $V_0 = \{x_1, \ldots, x_n\}$, and that we are given another finite set of free variables V_1, together with a map

$$V_0 \to V_1 \qquad (x_i \in V_0) \mapsto (y_i \in V_1).$$

We can then form a new formula with free variables in V_1 by replacing each occurrence of the variable x_i by y_i; we will denote this formula by $\varphi(y_1, \ldots, y_n)$. Note that we need not assume that the map $V_0 \to V_1$ is bijective. For example, if $\varphi(x, y)$ is a formula with free variables x and y, then we can consider $\varphi(z, z)$ as a formula with a single free variable z. We can also consider the case where V_0 is a subset of V_1 and each y_i is equal to x_i; in this case, we can regard $\varphi(\vec{x})$ as a formula with free variables in V_1, where it just happens that some of the variables have never been mentioned.

The preceding discussion comes with a caveat: for the substitution procedure $\varphi(x_1, \ldots, x_n) \mapsto \varphi(y_1, \ldots, y_n)$ to be sensible, we must assume that the variables of V_1 do not appear as *bound variables* in the formula φ. For example, if $\varphi(x)$ is a formula of the form $(\exists y)R(y, x)$, then it is sensible to write $\varphi(z)$ for the formula $(\exists y)R(y, z)$ for any variable z distinct from y, but we don't want to think of the sentence $(\exists y)R(y, y)$ as an instance of the formula φ.

Remark 4. We have opted for some economy here in the interest of keeping our definition compact. Using the preceding operations we can define several others. For example:

- We define $\varphi(\vec{x}) \wedge \varphi'(\vec{x})$ to be $\neg((\neg\varphi(\vec{x})) \vee (\neg\varphi'(\vec{x})))$,
- We define $\varphi(\vec{x}) \Rightarrow \varphi'(\vec{x})$ to be $(\neg\varphi(\vec{x})) \vee \varphi'(\vec{x})$.
- We define $\varphi(\vec{x}) \Leftrightarrow \varphi'(\vec{x})$ to be $(\varphi(\vec{x}) \Rightarrow \varphi'(\vec{x})) \wedge (\varphi'(\vec{x}) \Rightarrow \varphi(\vec{x}))$.
- Given a sequence of variables $\vec{y} = (y_1, \ldots, y_m)$ and a formula $\varphi(\vec{x}, \vec{y})$, we define $(\exists \vec{y})[\varphi(\vec{x}, \vec{y})]$ to be $(\exists y)(\exists y_2) \cdots (\exists y_m)[\varphi(\vec{x}, \vec{y})]$.
- Given a sequence of variables $\vec{y} = (y_1, \ldots, y_m)$ and a formula $\varphi(\vec{x}, \vec{y})$, we define $(\forall \vec{y})[\varphi(\vec{x}, \vec{y})]$ to be $\neg(\exists \vec{y})[\neg\varphi(\vec{x}, \vec{y})]$.
- Given a sequence of variables $\vec{y} = (y_1, \ldots, y_m)$ and a formula $\varphi(\vec{x}, \vec{y})$, we define $(\exists! \vec{y})[\varphi(\vec{x}, \vec{y})]$ to be

  $$(\exists \vec{y})[\varphi(\vec{x}, \vec{y})] \wedge (\forall \vec{y})(\forall \vec{z})[(\varphi(\vec{x}, \vec{y}) \wedge \varphi(\vec{x}, \vec{z})) \Rightarrow (y_1 = z_1) \wedge \cdots \wedge (y_m = z_m)]$$

  where $\vec{z} = (z_1, \ldots, z_m)$ is some list of auxiliary variables which are not bound in φ.

Let T be a first order theory. Our next goal is to introduce a category $\mathrm{Syn}_0(T)$, which we will refer to as the *weak syntactic category of* T. The objects of $\mathrm{Syn}_0(T)$ are just formulas in the language of T. To avoid confusion, if $\varphi(\vec{x})$ if a formula in the language of T, we will write $[\varphi(\vec{x})]$ for the corresponding object of $\mathrm{Syn}_0(T)$. Roughly speaking, we want to think of $[\varphi(\vec{x})]$ as the "collection of all tuples $\vec{x}$ satisfying φ". However, we view this as an abstract entity, existing without reference to any fixed model of T.

Definition 5 (Morphisms in $\mathrm{Syn}_0(T)$)**.** Let $X = [\varphi(\vec{x}]$ and $Y = [\psi(\vec{y})]$ be objects of $\mathrm{Syn}_0(T)$. A *morphism from X to Y* in $\mathrm{Syn}_0(T)$ consists of a collection of maps

$$f_M : M[\varphi] \to M[\psi],$$

defined for every model M of T, which are *definable* in the following sense:

($*$) After renaming the variables to arrange that the variables $\vec{x} = (x_1, \ldots, x_m)$ to not appear in ψ and the variables $\vec{y} = (y_1, \ldots, y_n)$ do not appear in φ, there exists a formula $\theta(\vec{x}, \vec{y})$ such that, for each model $M \vDash T$, we have

$$M[\theta] = \Gamma(f_M) := \{(c_1, \ldots, c_m, d_1, \ldots, d_n) : M \vDash \varphi(c_1, \ldots, c_m) \text{ and } f_M(c_1, \ldots, c_m) = (d_1, \ldots, d_n)\}$$

Proposition 6. *Let T be a first-order theory. Then:*

(*a*) *Let $X = [\varphi(\vec{x})]$ be an object of $\mathrm{Syn}_0(T)$. Then the collection of identity maps*

$$\{\mathrm{id}_{M[\varphi]} : M[\varphi] \to M[\varphi]\}_{M \vDash T}$$

is a morphism $\mathrm{id}_X : X \to X$ of $\mathrm{Syn}_0(T)$.

(*b*) *Let $X = [\varphi(\vec{x})]$, $Y = [\varphi'(\vec{y})]$, and $Z = [\varphi''(\vec{z})]$ be objects of $\mathrm{Syn}_0(T)$, and suppose we are given morphisms $X \xrightarrow{f} Y \xrightarrow{g} Z$ given by maps $f_M : M[\varphi] \to M[\varphi']$ and $g_M : M[\varphi'] \to M[\varphi'']$ for each model $M \vDash T$. Then the collection of composite maps $g_M \circ f_M$ determines a morphism from X to Z in $\mathrm{Syn}_0(T)$, which we will denote by $g \circ f$.*

Proof. To prove (*a*), we first choose some auxiliary variables $(y_1, \ldots, y_n)$ not appearing in $\varphi(\vec{x})$. We then observe that graph of the identity map $\mathrm{id} : M[\varphi] \to M[\varphi]$ is given by $M[\theta]$, where $\theta(\vec{x}, \vec{y})$ is the formula

$$\varphi(\vec{x}) \wedge (x_1 = y_1) \wedge \cdots \wedge (x_n = y_n).$$

To prove (*b*), we note that after suitable renaming of variables we may assume that there are formulas $\theta(\vec{x}, \vec{y})$ and $\theta'(\vec{y}, \vec{z})$ such that $M[\theta] = \Gamma(f_M)$ and $M[\theta'] = \Gamma(g_M)$ for each model $M \vDash T$. Then we have $\Gamma(g_M \circ f_M) = M[\rho]$, where $\rho(\vec{x}, \vec{z})$ is the formula $(\exists \vec{y})[\theta(\vec{x}, \vec{y}) \wedge \theta'(\vec{y}, \vec{z})]$. □

Corollary 7. *For every theory T, $\mathrm{Syn}_0(T)$ is a category (with the composition law described in Proposition 6). We will refer to $\mathrm{Syn}_0(T)$ as the weak syntactic category of T.*

Remark 8. Let T be a (typed) first-order theory and let φ be a sentence in the language of T. We will write $T \vDash \varphi$ if $M \vDash \varphi$ for every model M of T.

Let $X = [\varphi(\vec{x})]$ and $Y = [\psi(\vec{y})]$ be objects of $\mathrm{Syn}_0(T)$. Assume that the variables of $\vec{x}$ to not appear in ψ and the variables of $\vec{y}$ do not appear in φ (which we can always arrange by renaming variables). Unwinding the definitions, we see that the morphisms from X to Y in $\mathrm{Syn}_0(T)$ are given by equivalence classes of formulas $\theta(\vec{x}, \vec{y})$ such that

$$T \vDash (\forall \vec{x}, \vec{y})[\theta(\vec{x}, \vec{y}) \to \varphi(\vec{x}) \wedge \psi(\vec{y})] \wedge (\forall \vec{x})[\varphi(\vec{x}) \Rightarrow (\exists! \vec{y})\theta(\vec{x}, \vec{y})]$$

(More informally, this statement can be read as "T implies that θ defines the graph of a function from tuples $\vec{x}$ satisfying φ to tuples $\vec{y}$ satisfying ψ.") Here we consider two formulas $\theta(\vec{x}, \vec{y})$ and $\theta'(\vec{x}, \vec{y})$ to be equivalent if

$$T \vDash (\forall \vec{x}, \vec{y})[\theta(\vec{x}, \vec{y}) \Leftrightarrow \theta'(\vec{x}, \vec{y})].$$

More informally: θ and θ' are equivalent if T implies that they define the graph of the same function.

Warning 9. At this point, the reader might reasonably object that $\mathrm{Syn}_0(T)$ is defined by reference to the models of T, and is therefore not really "syntactic" in nature. To obtain a fully "syntactic" definition, we should define the objects of $\mathrm{Syn}_0(T)$ to be formulas in the language of T, with morphisms from $X = [\varphi(\vec{x})]$ and $Y = [\psi(\vec{y})]$ given by equivalence classes of formulas $\theta(\vec{x}, \vec{y})$ satisfying

$$T \vdash (\forall \vec{x}, \vec{y})[\theta(\vec{x}, \vec{y}) \to \varphi(\vec{x}) \wedge \psi(\vec{y})] \wedge (\forall \vec{x})[\varphi(\vec{x}) \Rightarrow (\exists! \vec{y})\theta(\vec{x}, \vec{y})],$$

where we consider $\theta(\vec{x}, \vec{y})$ and $\theta'(\vec{x}, \vec{y})$ to be equivalent if

$$T \vdash (\forall \vec{x}, \vec{y})[\theta(\vec{x}, \vec{y}) \Leftrightarrow \theta'(\vec{x}, \vec{y})].$$

To make this definition precise, we need to say what it means for a sentence φ to be provable in a theory T. After doing so, we ultimately discover that the result is equivalent to the definition we have given above, by virtue of Gödel's completeness theorem. However, this is in some sense "cheating": for example, later in this course we would like to apply these ideas to recover a proof of Gödel's theorem. We will return to this point later.

Construction 10. Let T be a (typed) first-order theory and let M be a model of T. For every object $X = [\varphi(\vec{x})]$ in the weak syntactic category $\mathrm{Syn}_0(T)$, we let $M[X]$ denote the set $M[\varphi] = \{(c_1, \ldots, c_n) : M \vDash \varphi(c_1, \ldots, c_n)\}$. It follows immediately from the definition of morphism in $\mathrm{Syn}_0(T)$ that we can regard the construction $X \mapsto M[X]$ as a functor from $\mathrm{Syn}_0(T)$ to the category $\mathcal{S}\mathrm{et}$ of sets.

Proposition 11. *Let T be a first-order theory and let M and N be models of T. The following data are equivalent:*

(1) *The datum of an elementary embedding $f_0 : M \to N$*

(2) *The datum of a natural transformation f from $M[\bullet]$ to $N[\bullet]$ (in the category of functors from $\mathrm{Syn}_0(T)$ to $\mathcal{S}\mathrm{et}$).*

Proof. Assume first that $f_0 : M \to N$ is an elementary embedding. For every formula $\varphi(x_1, \ldots, x_n)$ in the language of T and every $c_1, \ldots, c_n \in M$, we have

$$(M \vDash \varphi(c_1, \ldots, c_n)) \Leftrightarrow (N \vDash \varphi(f_0(c_1), \ldots, f_0(c_n))).$$

It follows that the construction $(c_1, \ldots, c_n) \mapsto (f_0(c_1), \ldots, f_0(c_n))$ determines a map of sets $f_\varphi : M[\varphi] \to N[\varphi]$. We claim that these maps determine a natural transformation of functors. In other words, for every morphism $g : [\varphi(x_1, \ldots, x_n)] \to [\psi(y_1, \ldots, y_m)]$ in the weak syntactic category, the diagram

$$\begin{array}{ccc} M[\varphi] & \xrightarrow{f_\varphi} & N[\varphi] \\ \downarrow{\scriptstyle g_M} & & \downarrow{\scriptstyle g_N} \\ M[\psi] & \xrightarrow{f_\psi} & N[\psi] \end{array}$$

commutes. Equivalently, we wish to show that the map $f_0^{n+m} : M^{n+m} \to N^{n+m}$ carries the graph $\Gamma(g_M)$ into the graph $\Gamma(g_N)$. Since g is a morphism in the syntactic category, we can choose a formula $\theta(\vec{x}, \vec{y})$ such that $\Gamma(g_m) = M[\theta]$ and $\Gamma(g_N) = N[\theta]$. We are therefore reduced to showing the implication

$$M \vDash \theta(c_1, \ldots, c_n, d_1, \ldots, d_m) \Rightarrow N \vDash \theta(f_0(c_1), \ldots f_0(c_n), f_0(d_1), \ldots f_0(d_m)),$$

which follows from our assumption that f_0 is an elementary embedding.

Now suppose that we are given a natural transformation of functors $f : M[\bullet] \to N[\bullet]$, given by a collection of maps $f_\varphi : M[\varphi] \to N[\varphi]$. Taking $\varphi(x)$ to be the formula $x = x$, we obtain a map $f_0 : M \to N$. We claim that f_0 is an elementary embedding. Let $\varphi(x_1, \ldots, x_n)$ be any formula in the language of T. It follows from the naturality of f that we have a commutative diagram

$$\begin{array}{ccc} M[\varphi] & \xrightarrow{f_\varphi} & N[\varphi] \\ \downarrow & & \downarrow \\ M^n & \xrightarrow{f_0^n} & N^n, \end{array}$$

where the vertical maps are inclusions. From the existence of such a diagram, we deduce the implication

$$(M \vDash \varphi(c_1, \ldots, c_n)) \Rightarrow (N \vDash \varphi(f_0(c_1), \ldots, f_0(c_n))).$$

Applying the same argument to the formula $\neg\varphi(\vec{x})$, we deduce the reverse implication, so that f_0 is an elementary embedding.

We leave it to the reader to verify that these constructions are inverse to one another. □

The conclusion of Proposition 11 is that it is not really necessary to distinguish between a model M of T and the associated functor $M[\bullet] : \mathrm{Syn}_0(T) \to \mathcal{S}\mathrm{et}$; they are essentially the same data. One can think of the functor $M[\bullet]$ as an "unbiased" presentation of the datum of a model of T: one does not think of a model as something that has a single underlying set: instead, one considers *all* the sets which can be constructed from the model by means of a first-order formula.

Of course, if we want to think about models as a special kind of functor, we will need to decide what kind of functors that we want to allow:

Question 12. Let T be a first-order theory and let $F : \mathrm{Syn}_0(T) \to \mathcal{S}\mathrm{et}$ be a functor. When does there exist a model M of T and an isomorphism of functors $F \simeq M[\bullet]$?

We will take up Question 12 in the next lecture.

We close by describing a mild generalization of the formalism described above. Recall that in the previous lecture, we introduced the notion of a *projective plane*. A projective plane is naturally described as a pair of sets P and L (the "points" and "lines", respectively), together with a binary relation between elements of P and elements of L. This is most naturally described by a slight variation on the preceding definitions.

Variant 13 (Typed First-Order Logic)**.** A *typed language* L consists of a set $\mathfrak{T}$ of *types* together with a set $\{P_i\}$ of predicate symbols. Each predicate symbol has an *arity* which is no longer a nonnegative integer, but a finite sequence $\vec{t} = (t_1, \ldots, t_n)$ of types.

We assume that each variable name is assigned a type, in such a way that there are infinitely many variable names of each type.

The notion of formula is defined as before, with two caveats:

(0) We add a new formula $\perp$ (for any number of free variables), to be understood simply as "false." This is necessary only in the case where there are no types (hence no variables); otherwise, we could just pick a variable x and replace $\perp$ by a contradictory assertion like $(\exists x)[\neg(x = x)]$.

(1′) We allow formulas of the form $x = y$ only when x and y are variables names of the same type.

(2′) If P_i is a predicate of arity $(t_1, \ldots, t_n)$, then we allow formulas of the form $P_i(x_1, x_2, \ldots, x_n)$ only when each x_j has type t_j.

An L-structure consists of a set $M[t]$ for every type $t \in \mathfrak{T}$, together with a subset $M[P_i] \subseteq M[t_1] \times \cdots \times M[t_n]$ for each predicate symbol P_i of arity $(t_1, \ldots, t_n)$. More generally, for any formula $\varphi(x_1, \ldots, x_n)$ with variables x_i of types t_i, we define the notion $M \vDash \varphi(c_1, \ldots, c_n)$ for $(c_1, \ldots, c_n) \in M[t_1] \times \cdots \times M[t_n]$ by induction as before, with the added clause that $M \vDash \perp (c_1, \ldots, c_n)$ is always false. We set

$$M[\varphi] = \{(c_1, \ldots, c_n) \in M[t_1] \times \cdots \times M[t_n] : M \vDash \varphi(c_1, \ldots, c_n)\}.$$

Example 14. In the special case where there is only one type, Variant 13 reduces to the definitions given above it.

Remark 15. In the case where there are only finitely many types, Variant 13 represents no real gain in generality. For example, instead of axiomatizing the theory of projective planes in terms of two sets P and L (together with the binary incidence relation $\subseteq P \times L$), one could consider the structure given by the disjoint union $P \amalg L$, augmented by two new unary relations

$$U(x) = \text{“}x \text{ is a point”} \qquad V(x) = \text{“}x \text{ is a line”}$$

$$(\forall x)(U(x) \vee V(x)) \qquad (\forall x)\neg[U(x) \wedge V(x)].$$

However, this sort of trick does not work if we want to allow infinitely many types.

Example 16 (Propositional Logic)**.** In the situation of Variant 13, suppose that the set $\mathfrak{T}$ of types is empty. Then the language L consists of a collection of symbols $\{P_i\}$, all of which are forced to have arity 0. The set V of variable names is empty (note that we still have infinitely many variables of each type, vacuously). All formulas are sentences, which can be built by combining the symbols P_i and $\bot$ by means of the negation operation $\neg$ and the disjunction operation $\vee$. An L-structure is equivalent to the datum of a map

$$\{P_i\} \to \{\text{True}, \text{False}\}$$

which assigns a truth value to each P_i. This determines, more generally, a truth value for each sentence φ built from the P_i.

Example 17 (The Syntactic Category of a Propositional Theory)**.** Let T be a propositional theory (that is, a typed first order theory with no types). Then every formula in the language of T is a sentence. Moreover, for every pair of sentences φ and ψ, we have

$$\text{Hom}_{\text{Syn}_0(T)}([\varphi], [\psi]) = \begin{cases} * & \text{if } T \vDash (\varphi \Rightarrow \psi) \\ \emptyset & \text{otherwise.} \end{cases}$$

In other words, the category $\text{Syn}_0(T)$ is equivalent to the partially ordered set of equivalence classes of sentences. This partially ordered set is a Boolean algebra, with join given by $(\varphi, \psi) \mapsto \varphi \vee \psi$ and complements given by $\varphi \mapsto \neg\varphi$.