CHAPTER 3

First(-order) things first (Cont'd)

2. S⌒e⌒m⌒a⌒n⌒t⌒i⌒c⌒s

2.1. Structures and assignments. Okay enough syntax for a minute, let's take a breath, and talk about what our formulas mean. In propositional logic, our sentences were always either true or false, but to make our formulas make sense, we needed to give each propositional variable a meaning, a so called assignment to either true or false. So the "place" in which propositional formulas take meaning is some universe that tells each variable if it should be T or F. Here, the goal is not to describe reasoning, but to describe mathematics, so kinda necessarily whatever this "universe" is, it will have to be a somewhat more complicated one.

Definition 2.1.1. Let $\mathcal{L}$ be a first-order language. An *$\mathcal{L}$-structure* $\mathcal{M}$ consist of a non-empty set M together with:

(1) An element $c^{\mathcal{M}} \in M$, for each constant symbol $\underline{c} \in \mathsf{Const}(\mathcal{L})$
(2) A subset $R^{\mathcal{M}} \subseteq M^n$, for each n-ary relation symbol $\underline{R} \in \mathsf{Rel}(R)$.
(3) A function $f^{\mathcal{M}} : M^n \to M$, for each n-ary function symbol $\underline{f} \in \mathsf{Fun}(\mathcal{L})$.

Altogether, we often write:

$$\mathcal{M} = \left(M; \left(c^{\mathcal{M}}\right)_{\underline{c}\in\mathsf{Const}(\mathcal{L})}, \left(R^{\mathcal{M}}\right)_{\underline{R}\in\mathsf{Rel}(\mathcal{L})}, \left(f^{\mathcal{M}}\right)_{\underline{f}\in\mathsf{Fun}(\mathcal{L})}\right).$$

We call M the *domain* or *universe* of the $\mathcal{L}$-structure $\mathcal{M}$. For each constant symbol $\underline{c} \in \mathsf{Const}(\mathcal{L})$ we call $c^{\mathcal{M}}$ the *interpretation* of $\underline{c}$ in $\mathcal{M}$ and similarly for relation symbols and function symbols.

Sometimes, we'll work with a language $\mathcal{L}$ we intuitively understand, such as $\mathcal{L}_{Peano}$, but of course in an $\mathcal{L}$-structure the interpretations of the symbols of the language can be as insane as we want. Let's do some examples to make sure we're on the same page:

Example 2.1.2. A pretty standard (this will be the punchline to a joke whose set up will appear several chapters later) $\mathcal{L}_{Peano}$-structure, $\mathcal{N}_{st}$ is the following:

- The universe of $\mathcal{N}_{st}$ is $\mathbb{N}$, the set of all natural numbers.
- The interpretation of the constant symbol $\underline{0}$ is the natural number 0.
- The interpretation of the unary function symbol $\underline{S}$ is the **successor function**:

$$\begin{aligned} \mathsf{succ} : \mathbb{N} &\to \mathbb{N} \\ x &\mapsto x+1 \end{aligned}$$

- The interpretation of the binary function symbols $\underline{+}$ and $\underline{\times}$ are the usual addition and multiplication functions, respectively.

Another example of an $\mathcal{L}_{Peano}$-structure, $\mathcal{N}_{looney}$ is the following:

- The universe of $\mathcal{N}_{looney}$ is $\omega^+ = \mathbb{N} \cup \{\mathbb{N}\}$, the successor ordinal of ω.
- The interpretation of the constant symbol $\underline{0}$ is the natural number 3.
- The interpretation of the unary function symbol $\underline{S}$ is a "loopy" function:

$$\begin{aligned} \mathsf{succ} : \omega^+ &\to \omega^+ \\ x &\mapsto \begin{cases} x+1 & \text{if } x \in \mathbb{N} \\ 0 & \text{if } x = \mathbb{N}. \end{cases} \end{aligned}$$

- The interpretation of the binary function symbols $\underline{+}$ and $\underline{\times}$ are constant functions that always return the number 9.

Both of these are very valid $\mathcal{L}_{Peano}$-structures.

Now, we know all about writing formulas, so given an $\mathcal{L}$-formula $\phi(x_1, \ldots, x_n)$ (remember, this means that ϕ is allowed to have free variables, and if it does, these are amongst $x_1, \ldots, x_n$) and an $\mathcal{L}$-structure $\mathcal{M} = (M; \ldots)$, we want to start understanding what ϕ could mean in $\mathcal{M}$. Of course, the meaning of ϕ should (and will) depend on the values that its variables take in M, just like in the case of propositional logic.

The next definition is, again, something we have seen before, in the context of propositional logic:

Definition 2.1.3. Let $\mathcal{L}$ be a first-order language and $\mathcal{M} = (M; \ldots)$ an $\mathcal{L}$-structure. An *assignment* is just a function α that assigns to each variable $x \in \mathsf{Var}$ an element of M, i.e. a function $\alpha : \mathsf{Var} \to M$.

So, for example, if the domain of an $\mathcal{L}$-structure is $\mathbb{N}$, and $\mathsf{Var} = \{x_0, x_1, \ldots\}$ (this is always possible, since we've assumed that our variables are a countable set, the use of other letters for variables is just to make things look pretty), then the following function is an assignment:

$$\alpha : \mathsf{Var} \to \mathbb{N}$$
$$x_i \mapsto i$$

I could almost give you, now, the definition of the semantics of first-order formulas, but first I'll need one more little notion. Given an assignment, we will define an important way of adjusting it, namely changing the value it gives to a single variable:

Notation 2.1.4. Let $\alpha : \mathsf{Var} \to M$ be an assignment, $x \in \mathsf{Var}$ and $b \in M$. We write $\alpha_{b/x}$ for the assignment:

$$\alpha_{b/x} : \mathsf{Var} \to M$$
$$y \mapsto \begin{cases} b & \text{if } x = y \\ \alpha(y) & \text{otherwise.} \end{cases}$$

More generally, let $x_1, \ldots, x_n \in \mathsf{Var}$ be pairwise distinct variables and $b_1, \ldots, b_n \in M$. Then we define:

$$\alpha_{b_1/x_1,\ldots,b_n/x_n} := (\alpha_{b_1/x_1,\ldots,b_{n-1}/x_{n-1}})_{b_n/x_n}.$$

This is another one of those inductive definitions. We change the values α gives to x_i, one at a time.

Great, that was abstract and seemingly useless. I feel like the best thing to do now is not to give you an example or whatever, but rather to keep throwing abstract definitions at you.

Definition 2.1.5. Let $\mathcal{L}$ be a first-order language, $\mathcal{M} = (M; \ldots)$ an $\mathcal{L}$-structure, and t an $\mathcal{L}$-term. Given an assignment $\alpha : \mathsf{Var} \to \mathcal{M}$, we define the *interpretation of t in $\mathcal{M}$ under α*, denoted $t^{\mathcal{M}}[\alpha]$, by induction, as follows:

(1) If t is the variable x, then $t^{\mathcal{M}}[\alpha]$ is the element $\alpha(x)$ of M.

(2) If t is the constant symbol $\underline{c}$, then $t^{\mathcal{M}}[\alpha]$ is the element $c^{\mathcal{M}}$ of M.

(3) If t is of the form $\underline{f}(t_1, \ldots, t_n)$, for an n-ary function symbol $f \in \mathsf{Fun}(\mathcal{L})$ and $\mathcal{L}$-terms $t_1, \ldots, t_n$, then $t^{\mathcal{M}}[\alpha]$ is the element $f^{\mathcal{M}}(t_1^{\mathcal{M}}[\alpha], \ldots, t_n^{\mathcal{M}}[\alpha])$ of M.

Example 2.1.6. Recall our $\mathcal{L}_{Peano}$-structure $\mathcal{N}_{st}$ from before, and the assignment $\alpha : \mathsf{Var} \to \mathbb{N}$. The interpretation under α of the term $x_1 \underline{\times} x_2$ in $\mathcal{N}_{st}$ is just the natural number 2.

To make life a little bit more complicated, recall our other $\mathcal{L}_{Peano}$-structure, $\mathcal{N}_{looney}$. An interpretation for this structure is just a map $\mathsf{Var} \to \omega^+$, and since Var is countable, we can just assume that $\mathsf{Var} = \{x_1, x_2, \dots\} \cup \{x_\omega\}$. Take α to be the function that maps x_i to i (where i is allowed to be $\mathbb{N}$). Then, the interpretation of $x_1 \underline{\times} x_\omega$ is the natural number 9. Remember, $\underline{\times}$ in this structure was the constant function that always returned the number 9. The interpretation of $\underline{S}(\underline{0})$ is the natural number 10, and the interpretation of $\underline{S}(x_\omega)$ is the natural number 0.

Here's a lemma that should also look familiar:

Lemma 2.1.7. *Let $\mathcal{L}$ be a first-order language, $\mathcal{M}$ an $\mathcal{L}$-structure and t an $\mathcal{L}$-term. Let α and β be two assignments such that for all $x \in \mathsf{Var}(t)$ we have $\alpha(x) = \beta(x)$. Then, $t^{\mathcal{M}}[\alpha] = t^{\mathcal{M}}[\beta]$.*

Proof. We prove this by induction on the length of terms:

- If t is the variable x, then, by assumption we have that $\alpha(x) = \beta(x)$, and thus $t^{\mathcal{M}}[\alpha] = \alpha(x) = \beta(x) = t^{\mathcal{M}}[\beta]$.
- If t is a constant symbol $\underline{c}$ then $t^{\mathcal{M}}[\alpha] = c^{\mathcal{M}} = t^{\mathcal{M}}[\beta]$.
- Suppose that t is of the form $\underline{f}(t_1, \dots, t_n)$ for some n-ary function symbol $\underline{f}$ and $\mathcal{L}$-terms $t_1, \dots, t_n$. Given two assignments $\alpha, \beta : \mathsf{Var} \to \mathcal{M}$, if they agree on all the variables in t, then they agree on all the variables of each t_i, for $i \leq n$. Thus, by induction $t_i^{\mathcal{M}}[\alpha] = t_i^{\mathcal{M}}[\beta]$, for $i \leq n$. So:
  $$t^{\mathcal{M}}[\alpha] = f^{\mathcal{M}}(t_1^{\mathcal{M}}[\alpha], \dots, t_n^{\mathcal{M}}[\alpha]) = f^{\mathcal{M}}(t_1^{\mathcal{M}}[\beta], \dots, t_n^{\mathcal{M}}[\beta]) = t^{\mathcal{M}}[\beta],$$
  and we're done. □

We will now extend Notation 1.2.20 to terms, in a natural way:

Notation 2.1.8. If t is an $\mathcal{L}$-term with variables amongst $x_1, \dots, x_n$, then we may denote this by $t(x_1, \dots, x_n)$.[1]

2.2. Truth. We have seen how to use assignments to give meaning to terms. Let's put all of this together, to define what it means for a formula to be true in a structure.

Definition 2.2.1 (Tarski's definition of truth)**.** Let $\mathcal{L}$ be a first-order language, $\mathcal{M} = (M; \dots)$ an $\mathcal{L}$-structure, and ϕ be an $\mathcal{L}$-formula. Let $\alpha : \mathsf{Var} \to M$ be an

[1] Recall that all variables in a term are free.

assignment. We define what it means for ϕ *to be satisfied in* $\mathcal{M}$*, under the assignment* α, denoted $\mathcal{M} \vDash \phi[\alpha]$, by induction on the structure of ϕ, as follows:

(1) If ϕ is an atomic formula, then there are two cases to consider:
- $\mathcal{M} \vDash (t_1 \doteq t_2)[\alpha]$ if and only if $t_1^{\mathcal{M}}[\alpha] = t_2^{\mathcal{M}}[\alpha]$, for $\mathcal{L}$-terms t_1 and t_2
- $\mathcal{M} \vDash \underline{R}(t_1, \dots, t_n)[\alpha]$ if and only if $R^{\mathcal{M}}(t_1^{\mathcal{M}}[\alpha], \dots, t_n^{\mathcal{M}}[\alpha])$, for $\underline{R} \in \mathsf{Rel}(\mathcal{L})$ an n-ary relation symbol and $\mathcal{L}$-terms $t_1, \dots, t_n$.

(2) If we have defined $\mathcal{M} \vDash \phi_1[\alpha]$ and $\mathcal{M} \vDash \phi_2[\alpha]$, then we set:
(a) $\mathcal{M} \vDash (\phi_1 \wedge \phi_2)[\alpha]$ if and only if $\mathcal{M} \vDash \phi_1[\alpha]$ and $\mathcal{M} \vDash \phi_2[\alpha]$.
(b) $\mathcal{M} \vDash (\phi_1 \vee \phi_2)[\alpha]$ if and only if $\mathcal{M} \vDash \phi_1[\alpha]$ or $\mathcal{M} \vDash \phi_2[\alpha]$.
(c) $\mathcal{M} \vDash (\neg\phi_1)[\alpha]$ if and only if it is not the case that $\mathcal{M} \vDash \phi[\alpha]$.
(d) $\mathcal{M} \vDash (\phi_1 \to \phi_2)[\alpha]$ if and only if [the obvious thing happens].
(e) $\mathcal{M} \vDash (\forall x)\phi[\alpha]$ if and only if for all $b \in M$ we have that $\mathcal{M} \vDash \phi_1[\alpha_{b/x}]$.
(f) $\mathcal{M} \vDash (\exists x)\phi[\alpha]$ if and only if for some $b \in M$ we have that $\mathcal{M} \vDash \phi_1[\alpha_{b/x}]$.

We write $\mathcal{M} \nvDash \phi[\alpha]$ as shorthand for: "it is not the case that $\mathcal{M} \vDash \phi[\alpha]$".[2]

This definition seems rather void of content. But it's really the core of this whole thing. The content of the definition is that the syntax we came up to express stuff has the meaning we want it to have. In all of the above cases, we are giving real meaning to a purely syntactic relation.

Example 2.2.2. Here's a very silly example. You'll be asked to do more examples, most of them less silly in HW3. In our structure $\mathcal{N}_{looney}$, for any assignment α, we have that:

$$\mathcal{N}_{looney} \vDash x_1 \underline{\times} x_2 \doteq \underline{S}(\underline{S}(\underline{S}(\underline{S}(\underline{S}(\underline{S}(\underline{0}))))))[\alpha].$$

What assignments do in $\mathcal{N}_{st}$ is much more interesting and complicated.

Just like when we did truth tables the truth tables themselves didn't say much, this is the grown-up version of truth tables (indeed, the first four clauses are precisely the truth tables from propositional logic). Let's do some logic!

Lemma 2.2.3. *Let* $\mathcal{M}$ *be an* $\mathcal{L}$*-structure and* ϕ *an* $\mathcal{L}$*-formula. Then, the following are equivalent:*

[2] Immediately from the definition, we have that $\mathcal{M} \vDash \neg\phi[\alpha]$ if and only if $\mathcal{M} \nvDash \phi[\alpha]$.

(1) $\mathcal{M} \vDash \phi[\alpha]$

(2) $\mathcal{M} \nvDash \neg\phi[\alpha]$

(3) $\mathcal{M} \vDash \neg\neg\phi[\alpha]$

for any assignment α.

This will be a bit of a word-salad, but it does have some moral point, it tells us that for any assignment α we can replace the syntactic object ϕ by the syntactic object $\neg\neg\phi$, and the semantic notions of satisfaction will be unaffected, which (unless you're an intuit) is a great thing.

PROOF. The equivalence of (2) and (3) is by (the footnote in the) definition. We show that (1) and (2) are equivalent.

We have to prove this by induction on the structure of ϕ. Suppose that ϕ is atomic. Then:

(1) Suppose that ϕ is of the form $t_1 \doteq t_2$ for $\mathcal{L}$-terms t_1 and t_2. Then:

$$\begin{aligned}\mathcal{M} \vDash (t_1 \doteq t_2)[\alpha] &\text{ iff } t_1^{\mathcal{M}}[\alpha] = t_2^{\mathcal{M}}[\alpha]\\ &\text{ iff it is not the case that } t_1^{\mathcal{M}}[\alpha] \neq t_2^{\mathcal{M}}[\alpha]\\ &\text{ iff it is not the case that it is not the case that } t_1^{\mathcal{M}}[\alpha] = t_2^{\mathcal{M}}[\alpha]\\ &\text{ if and only if it is not the case that } \mathcal{M} \vDash \neg(t_1 \doteq t_2)[\alpha]\\ &\text{ iff } \mathcal{M} \nvDash \neg(t_1 \doteq t_2)[\alpha].\end{aligned}$$

(2) Suppose that ϕ is of the form $\underline{R}(t_1, \dots, t_n)$ for an n-ary relation symbol $\underline{R} \in \mathsf{Rel}(\mathcal{L})$ and $\mathcal{L}$-terms $t_1, \dots, t_n$. Then:

$$\begin{aligned}\mathcal{M} \vDash \underline{R}(t_1, \dots, t_n)[\alpha] &\text{ iff } R^{\mathcal{M}}(t_1^{\mathcal{M}}[\alpha], \dots, t_n^{\mathcal{M}}[\alpha])\\ &\text{ iff it's not the case that it's not the case that } R^{\mathcal{M}}(t_1^{\mathcal{M}}[\alpha], \dots, t_n^{\mathcal{M}}[\alpha])\\ &\text{ iff } \mathcal{M} \nvDash \neg\underline{R}(t_1, \dots, t_n)[\alpha].\end{aligned}$$

Now, for the inductive part. Suppose that we have shown the result for formulas ϕ_1 and ϕ_2 (and any assignments).

Exercise 2.2.4. Show that if $\mathcal{M} \vDash (\phi_1 \wedge \phi_2)[\alpha]$ if and only if $\mathcal{M} \nvDash (\neg(\phi_1 \wedge \phi_2))[\alpha]$. Then, do the analogous things for $\vee$, $\rightarrow$ and $\neg$.

Suppose that ϕ is of the form $(\forall x)\phi_1$. Then:

$\mathcal{M} \models (\forall x)\phi_1[\alpha]$ iff for all $b \in M$ we have $\mathcal{M} \models \phi_1[\alpha_{b/x}]$
iff for all $b \in M$ we have $\mathcal{M} \not\models \neg\phi_1[\alpha_{b/x}]$
iff for all $b \in M$ it is not the case that $\mathcal{M} \models \neg\phi_1[\alpha_{b/x}]$
iff it's not the case that for some $b \in M$ we have $\mathcal{M} \models \neg\phi_1[\alpha_{b/x}]$
iff it's not the case that for some $b \in M$ it is not the case that $\mathcal{M} \models \phi_1[\alpha_{b/x}]$
iff it's not the case that it's not the case that for all $b \in M$ we have $\mathcal{M} \models \phi_1[\alpha_{b/x}]$
iff it's not the case that $\mathcal{M} \not\models (\forall x)\phi_1$
iff it's not the case that $\mathcal{M} \models \neg(\forall x)\phi_1$
iff $\mathcal{M} \not\models \neg(\forall x)\phi$,

which is what we wanted to show.

Exercise 2.2.5. Do the same when ϕ is of the form $(\exists x)\phi_1$.

Having done all the cases, the result follows. □

I mean it probably shouldn't feel like it not the case that something happened, or it should. Anyway it should feel like nothing happened. Here's more nothing:

THEOREM 2.2.6. *Let $\mathcal{M}$ be an $\mathcal{L}$-structure and ϕ an $\mathcal{L}$-formula. Then, the following are equivalent:*

(1) $\mathcal{M} \models (\forall x)\phi[\alpha]$

(2) $\mathcal{M} \models \neg(\exists x)(\neg\phi)[\alpha]$

for any assignment α

PROOF. We have that:

$\mathcal{M} \models (\forall x)\phi[\alpha]$ iff for all $b \in M$ we have that $\mathcal{M} \models \phi[\alpha_{b/x}]$
iff for all $b \in M$ we have that $\mathcal{M} \not\models \neg\phi[\alpha_{b/x}]$
iff there is no $b \in M$ for which we have $\mathcal{M} \models \neg\phi[\alpha_{b/x}]$
iff it is not the case that there is $b \in M$ s.t. $\mathcal{M} \models \neg\phi[\alpha_{b/x}]$
iff it is not the case that $\mathcal{M} \models (\exists x)\neg\phi[\alpha]$
iff $\mathcal{M} \models \neg(\exists x)(\neg\phi)$,

which is what we wanted to show. □

Let's see something we have seen before:

THEOREM 2.2.7. *Let $\phi(x_1, \ldots, x_n)$ be an $\mathcal{L}$-formula and $\mathcal{M}$ an $\mathcal{L}$-structure. For any assignments α, β such that for all $i \leq n$ we have that $\alpha(x_i) = \beta(x_i)$, we have that $\mathcal{M} \vDash \phi[\alpha]$ if and only if $\mathcal{M} \vDash \phi[\beta]$.*

PROOF. The proof is by induction on ϕ. I will only write down the case when ϕ is of the form $(\forall x)\psi$ (see next exercise). Let α and β be as above. Let $c \in M$ be any element. Observe that if α and β agree on all free variables of ϕ, then so do $\alpha_{c/x}$ and $\beta_{c/x}$. Thus:

$$\begin{aligned} \mathcal{M} \vDash (\forall x)\psi[\alpha] &\text{ iff for all } c \in M \ \mathcal{M} \vDash \psi[\alpha_{c/x}] \\ &\text{ iff for all } c \in M \ \mathcal{M} \vDash \psi[\beta_{c/x}] \\ &\text{ iff } \mathcal{M} \vDash (\forall x)\psi[\beta]. \end{aligned}$$

□

Exercise 2.2.8. Write out the full proof of the theorem above (you don't have to rewrite the part of the proof that I wrote up, but you should certainly discuss what happens with terms and atomic formulas).

Similarly to propositional sentences, we have the following:

Corollary 2.2.9. *If ϕ is an $\mathcal{L}$-sentence, then for any $\mathcal{L}$-structure $\mathcal{M}$ we have that either:*

(1) For any assignment $\alpha : \mathsf{Var} \to \mathcal{M}$ we have that $\mathcal{M} \vDash \phi[\alpha]$.

(2) For any assignment $\alpha : \mathsf{Var} \to \mathcal{M}$ we have that $\mathcal{M} \nvDash \phi[\alpha]$.

Thus, if ϕ is a sentence, we are justified in writing $\mathcal{M} \vDash \phi$ to indicate that we are in case *(1)* above.

3. Syntax and semantics don't always play well together

We'll now get deeper in the weeds of first-order logic. Our goal is to explore the interplay of syntax and semantics a little bit better. First of all, though we need to fix some more semantic definitions. The first subsection is rather small, but it introduces some of the most important definitions of the course.

3.1. Up to logical equivalence. Now that we have semantics, let's start naming things. First and foremost:

An $\mathcal{L}$**-theory** is just a set of $\mathcal{L}$-sentences.

For example $\{(\forall x)(x \neq x)\}$ is an $\mathcal{L}$-theory (in any language $\mathcal{L}$). So is:

$$\{\forall x \underline{R}(x,x), \forall x \forall y \forall z(\underline{R}(x,y) \wedge \underline{R}(y,z) \to \underline{R}(x,z)), \forall x \forall yz(\underline{R}(x,y) \to \underline{R}(y,x))\},$$

in a language with a binary relation symbol $\underline{R}$. Of course, $\mathcal{L}$-theories are allowed to be infinite, these are harder to write down, but here's an example:

$$\left\{ \exists x_1 \dots \exists x_n \bigwedge_{1 \leq i < j \leq n} x_i \neq x_j : n \in \mathbb{N} \right\}.$$

Definition 3.1.1. We say that an $\mathcal{L}$-structure $\mathcal{M}$ is a **model** of an $\mathcal{L}$-theory T if, for all $\phi \in T$ we have that $\mathcal{M} \vDash \phi$. We denote this by $\mathcal{M} \vDash T$.

Exercise 3.1.2. Find a model of each of the previous three theories (in an appropriate language of your choice).

Ha! If you solved the previous exercise, you may have seen that the first theory did not have a model. We have a word for theories like that one:

Definition 3.1.3. We say that an $\mathcal{L}$-theory T is *satisfiable* if there is an $\mathcal{L}$ structure $\mathcal{M}$ such that $\mathcal{M} \vDash T$.

We further extend some definitions we saw in propositional logic, this time for $\mathcal{L}$-sentences, follows:

- We say that ϕ is a **logical consequence** of ψ if whenever $\mathcal{M} \vDash \phi$ we have that $\mathcal{M} \vDash \psi$. More generally, if T is an $\mathcal{L}$-theory, we overload this notation, and write $T \vDash \phi$ to mean that every model of T is a model of ϕ.
- We say that two sentences are **logically equivalent** if each is a logical consequence of the other.

In the next little subsubsection, I'll discuss a way of viewing predicate logic inside first-order logic. It's not at all elegant, and this is the point where I totally regret not introducing Boolean algebras, but here goes nothing...

3.1.1. *Propositional in first.* Before we need to do more syntax, let's take a breath and a step back. Let $\mathcal{L}$ be a language with no function and no relation symbols and two constant symbols $\top$ and $\bot$. Let T be the $\mathcal{L}$-theory:

$$\{\neg(\top \doteq \bot)\} \cup \{\forall x \forall y \forall z (z = x \vee z = y)\}$$

and let $\mathcal{M} = (M; \top^{\mathcal{M}}, \bot^{\mathcal{M}})$ be an $\mathcal{L}$-structure. If $\mathcal{M} \vDash T$, then M has exactly two elements. An interpretation is just a map sending $\top$ to $\top^{\mathcal{M}}$ and $\bot$ to $\bot^{\mathcal{M}}$ where $\top^{\mathcal{M}} \neq \bot^{\mathcal{M}}$.

Given a propositional formula ϕ with propositional variables $A_1, \ldots, A_n$, we construct an $\mathcal{L}$-formula $\phi_{prp}(x_1, \ldots, x_n)$ by replacing each variable A_i in ϕ with the atomic $\mathcal{L}$-formula $x_i = \top$ and replacing every instance of $\top$ in ϕ with the atomic $\mathcal{L}$-formula $\top = \top$ and every instance of $\bot$ in ϕ with the atomic $\mathcal{L}$-formula $\top = \bot$.

Given a (propositional) assignment $\mathcal{A} : \mathsf{Var} \to \{T, F\}$ define a (first-order) assignment $\alpha_{\mathcal{A}} : \mathsf{Var} \to \mathcal{M}$ by setting $\alpha_{\mathcal{A}}(x_i) = \top^{\mathcal{M}}$ if and only if $\mathcal{A}(A_i) = T$.

Then we have that:

- For any propositional assignment $\mathcal{A}$, we have that $\mathcal{A} \vDash \phi$ if and only if $\mathcal{M} \vDash \phi_{prp}[\alpha_{\mathcal{A}}]$.
- A propositional formula ϕ is a propositional tautology if and only if $\mathcal{M} \vDash (\forall x_1) \cdots (\forall x_n) \phi_{prp}$

Exercise 3.1.4. Prove the two points above.

End of digression

We'll get back to connections between propositional and first-order logic later. For now I want to discuss something rather annoying.

3.2. Syntax can be annoying II: Electric Substitutionaloo. We've cleaned up our formulas but things can still go wrong. Remember, we defined previously what it means to substitute a variable for a different one. Now, we'll see what happens when we try to substitute terms for variables. For closed terms, this is kind of obvious, we just put them where we should put them and go on with our day. For terms which contain variables though the waters murk up quickly.

The usual example people give for this is the following:

Example 3.2.1. Let $\mathcal{L}$ be any first-order language and consider the following $\mathcal{L}$-formula $\phi(y)$:

$$(\exists x)\neg(x \doteq y)$$

For any $\mathcal{L}$-structure $\mathcal{M} = (M; \dots)$ with at least two elements in its universe and any interpretation $\alpha : \mathsf{Var} \to M$ we see that:

$$\mathcal{M} \vDash \phi[\alpha].$$

This is the same if we substitute for y any variable z different from x. However, if we consider $\phi[x/y]_{free}$, then we end up with the formula:

$$(\exists x)\neg(x \doteq x)$$

which is never satisfied in ANY structure.

What we'd like is a notion of substitution that is not affected by this. More explicitly, we'd like a way of computing $\phi[x/y]$ which results in a formula such that:

For every assignment α we have that $\mathcal{M} \vDash \phi[\alpha]$ if and only if

$$\mathcal{M} \vDash (\phi[x/y])[\alpha_{\alpha(y)/x}]$$

That is, we'd like to not care about bound variables and have a way of substitution that respects assignments.

Why are we going through this trouble, you may be asking yourselves. The real answer is that just like in the case of propositional logic we will eventually prove a theorem connecting syntax and semantics,[3] and (a) as we saw in the definition of truth, substitutions are crucial to evaluate truth, because we want a formal proof system (i.e. a computerised procedure to produce proofs) we need to be able to trust that our computer can perform substitutions (SYNTAX) without changing the meaning of the things its proving (SEMANTICS). The way we'll get to it below is not the only possible, but it works.

Definition 3.2.2. Let $\mathcal{L}$ be a first-order language, t an $\mathcal{L}$-term $x_1, \dots, x_n \in \mathsf{Var}$ distinct variables and $s_1, \dots, s_n$ be $\mathcal{L}$-terms. We define the term $t[\bar{s}/\bar{x}]$ inductively, as follows:

(1) If t is the variable y then:

$$y[\bar{s}/\bar{x}] = \begin{cases} y & \text{if } y \notin \{x_1, \dots, x_n\} \\ s_i & \text{if } y = x_i \end{cases}$$

[3]The Soundness and Completeness Theorem of course.

(2) If t is a constant symbol, then $t[\bar{s}/\bar{x}]$ is just t.

(3) If t is of the form $\underline{f}(t_1, \ldots, t_n)$ for some n-ary function symbol $\underline{f} \in \mathsf{Fun}(\mathcal{L})$ and $\mathcal{L}$-terms $t_1, \ldots, t_n$, then $t[\bar{s}/\bar{x}]$ is the term $\underline{f}\,(t_1[\bar{s}/\bar{x}], \ldots, t_n[\bar{s}/\bar{x}])$.

The first two parts of the previous definition were just like the first two parts of Definition 1.2.7, but where things started next exercise is here to convince you that we do indeed need to do our substitutions simultaneously and not sequentially:

Exercise 3.2.3. Let $\mathcal{L}$ be a language with a binary function symbol $\underline{f}$. Write down an example of an $\mathcal{L}$-term t and $\mathcal{L}$-terms $s_1, \ldots, s_n$ such that $t[s_1/x_1, \ldots, s_n/x_n] \neq t[\bar{s}/\bar{x}]$.

Now, we continue to atomic formulas in the obvious way:

Definition 3.2.4. Now, let ϕ be an atomic $\mathcal{L}$-formula, $x_1, \ldots, x_n \in \mathsf{Var}$ distinct variables and $s_1, \ldots, s_n$ be $\mathcal{L}$-terms. Then, the atomic formula $\phi[\bar{s}/\bar{x}]$ is defined inductively as follows

(1) If ϕ is of the form $t_1 \doteq t_2$ for $\mathcal{L}$-terms t_1, t_2 then $\phi[\bar{s}/\bar{x}]$ is the formula $t_1[\bar{s}/\bar{x}] \doteq t_2[\bar{s}/\bar{x}]$.

(2) If ϕ is of the form $\underline{R}(t_1, \ldots, t_n)$ for some n-ary relation symbol $\underline{R} \in \mathsf{Rel}(\mathcal{L})$ and $\mathcal{L}$-terms $t_1, \ldots, t_n$, then $\phi[\bar{s}/\bar{x}]$ is the formula $\underline{R}(t_1[\bar{s}/\bar{x}], \ldots, t_n[\bar{s}/\bar{x}])$.

Finally:

Definition 3.2.5. Let $\mathcal{L}$ be a first-order language, ϕ an $\mathcal{L}$-formula, $x_1, \ldots, x_n \in \mathsf{Var}$ distinct variables and $s_1, \ldots, s_n$ be $\mathcal{L}$-terms. We define $\phi[\bar{s}/\bar{x}]$ inductively, as follows:

(1) If ϕ is atomic, then $\phi[\bar{s}/\bar{x}]$ is just the atomic formula $\phi[y/x]$ we defined above.

(2) Suppose we have defined $\phi[\bar{s}/\bar{x}]$, $\phi_1[\bar{s}/\bar{x}]$ and $\phi_2[\bar{s}/\bar{x}]$. Then:

- $(\phi_1 \wedge \phi_2)[\bar{s}/\bar{x}]$ is the formula $(\phi_1[\bar{s}/\bar{x}] \wedge \phi_2[\bar{s}/\bar{x}])$.
- $(\phi_1 \vee \phi_2)[\bar{s}/\bar{x}]$ is the formula $(\phi_1[\bar{s}/\bar{x}] \vee \phi_2[\bar{s}/\bar{x}])$.
- $(\phi_1 \rightarrow \phi_2)[\bar{s}/\bar{x}]$ is the formula $(\phi_1[\bar{s}/\bar{x}] \rightarrow \phi_2[\bar{s}/\bar{x}])$.
- $(\neg\phi_1)[\bar{s}/\bar{x}]$ is the formula $(\neg\phi_1[\bar{s}/\bar{x}])$.
- $(\forall y)\phi[\bar{s}/\bar{x}]$ is harder. Let $\{x_{i_1}, \ldots, x_{i_k}\} = \mathsf{Free}((\forall y)\phi) \cap \{x_1, \ldots, x_n\}$. We consider two cases:

- Case 1: If $y \notin \bigcup_{1\leq i\leq n} \mathsf{Var}(s_i)$ then

$$(\forall y)\phi[\bar{s}/\bar{x}] = (\forall y)(\phi[s_{i_1}, \dots s_{i_k}/x_{i_1}, \dots, x_{i_k}])$$

- Case 2. If $y \in \bigcup_{1\leq i\leq n} \mathsf{Var}(s_i)$ then:

$$(\forall y)\phi[\bar{s}/\bar{x}] = (\forall z)(\phi[s_{i_1}, \dots s_{i_k}, z/x_{i_1}, \dots, x_{i_k}, y])$$

- $(\exists y)\phi[\bar{s}/\bar{x}]$ is defined similarly.

Exercise 3.2.6. Write out the definition of $(\exists y)\phi[\bar{s}/\bar{x}]$.

The main result of this section is the following:

THEOREM 3.2.7 (The Substitution Lemma). *Let $x_1, \dots, x_n$ be distinct variables, $s_1, \dots, s_n$ terms and $\alpha : \mathsf{Var} \to \mathcal{M}$ an assignment. For every first-order formula ϕ we have that:*

$$\mathcal{M} \vDash (\phi[\bar{s}/\bar{x}])[\alpha] \text{ if and only if } \mathcal{M} \vDash \phi\left[\alpha_{(s_1^{\mathcal{M}}[\alpha])/x_1, \dots, (s_n^{\mathcal{M}}[\alpha])/x_n}\right]$$

Yeah, you're right, I should explain what is happening here. This is really the desideratum I put forward at the beginning of this whole story.

Recall that if s is a term and α is an assignment then $s^{\mathcal{M}}[\alpha]$ is an element of M (namely the element where all variables in s have been assigned values by α and all non-logical symbols in s are interpreted as $\mathcal{M}$ intended.

So, for our given assignment α, each of $s_1^{\mathcal{M}}[\alpha], \dots, s_n^{\mathcal{M}}[\alpha]$ is an element of M. Moreover, we have a way of changing up assignments at specific values, explicitly,

$$\alpha_{s_1^{\mathcal{M}}[\alpha]/x_1, \dots, s_n^{\mathcal{M}}[\alpha]/x_n}(y) := \begin{cases} \alpha(y) & \text{if } y \notin \{x_1, \dots, x_n\} \\ s_i^{\mathcal{M}}[\alpha] & \text{if } y = x_i. \end{cases}$$

So this assignment here first evaluates each of the terms s_i according to the α we started with and then replaces the value that α gives to each x_i with the value it computed for s_i.

So the RHS says: Compute the values that the terms s_i take under α and then substitute all the variables in ϕ plugging in these terms where appropriate. The LHS says: Substitute the variables in ϕ with the terms s_i and then evaluate using α. As we saw before, if we're not careful about our substitutions, this need not be the case.

PROOF OF THE SUBSTITUTION LEMMA. First, we show that if T is an $\mathcal{L}$-term then:

$$(t[\bar{s}/\bar{x}])^{\mathcal{M}}[\alpha] = t^{\mathcal{M}}[\alpha_{s_1^{\mathcal{M}}[\alpha]/x_1,\ldots,s_n^{\mathcal{M}}[\alpha]/x_n}]$$

This is the usual case-by-case induction argument:

- If t is a variable y we have

$$\begin{aligned}(y[\bar{s}/\bar{x}])^{\mathcal{M}}[\alpha] &= \begin{cases}\alpha(y) & \text{if } y \notin \{x_1,\ldots,x_n\}\\ s_i^{\mathcal{M}}[\alpha] & \text{if } y = x_i\end{cases}\\ &= y^{\mathcal{M}}[\alpha_{s_1^{\mathcal{M}}[\alpha]/x_1,\ldots,s_n^{\mathcal{M}}[\alpha]/x_n}]\end{aligned}$$

- If t is the constant symbol $\underline{c}$, then:

$$(t[\bar{s}/\bar{x}])^{\mathcal{M}}[\alpha] = c^{\mathcal{M}} = t^{\mathcal{M}}[\alpha_{s_1^{\mathcal{M}}[\alpha]/x_1,\ldots,s_n^{\mathcal{M}}[\alpha]/x_n}]$$

- If t is of the form $\underline{f}(t_1,\ldots,t_m)$, for an m-ary function symbol $\underline{f} \in \mathsf{Fun}(\mathcal{L})$ and terms $t_1,\ldots,t_m$, then we have:

$$\begin{aligned}(\underline{f}(t_m,\ldots,t_m)[\bar{s}/\bar{x}])^{\mathcal{M}}[\alpha] &= f\left((t_1[\bar{s}/\bar{x}])^{\mathcal{M}}[\alpha],\ldots,(t_m[\bar{s}/\bar{x}])^{\mathcal{M}}[\alpha]\right)\\ &= f\left((t_m)^{\mathcal{M}}[\alpha_{s_1^{\mathcal{M}}[\alpha]/x_1,\ldots,s_n^{\mathcal{M}}[\alpha]/x_n}],\ldots,(t_m)^{\mathcal{M}}[\alpha_{s_1^{\mathcal{M}}[\alpha]/x_1,\ldots,s_n^{\mathcal{M}}[\alpha]/x_n}]\right)\\ &= \underline{f}(t_1,\ldots,t_m)^{\mathcal{M}}[\alpha_{s_1^{\mathcal{M}}[\alpha]/x_1,\ldots,s_n^{\mathcal{M}}[\alpha]/x_n}]\end{aligned}$$

Now for atomic formulas:

- $t_1 \doteq t_2$ is immediate from the above.
- $\underline{R}(t_1,\ldots,t_m)$ – See next exercise.

Finally, for formulas proper, there are a few cases to consider. I'll only do the hard case and leave the rest as exercise.

- $\phi_1 \wedge \phi_2$, $\phi_1 \vee \phi_2$, $\phi_1 \to \phi_2$, $\neg\phi_1$, and $(\exists y)\phi$ – See next exercise.
- $(\forall y)\phi$. Let $\{x_{i_1},\ldots,x_{i_k}\} = \mathsf{Free}((\forall y)\phi) \cap \{x_1,\ldots,x_n\}$ There are two cases to consider:
- Case 1: If $y \notin \bigcup_{1\le i\le n}\mathsf{Var}(s_i)$ then, by definition:

$$(\forall y)\phi[\bar{s}/\bar{x}] = (\forall y)(\phi[s_{i_1},\ldots,s_{i_k}/x_{i_1},\ldots,x_{i_k}])$$

In particular, we have that:

$$\begin{aligned}
\mathcal{M} \vDash ((\forall y)\phi[\bar{s}/\bar{x}])[\alpha] &\text{ iff for all } b \in M \text{ we have } \mathcal{M} \vDash (\phi[\bar{s}/\bar{x}])[\alpha_{b/y}] \\
&\text{iff for all } b \in M \text{ we have } \mathcal{M} \vDash \phi\left[\alpha_{(s_{i_1}^{\mathcal{M}}[\alpha_{b/y}])/x_{i_1},\ldots,(s_{i_k}^{\mathcal{M}}[\alpha_{b/y}])/x_{i_k}}\right] \\
&\text{iff } \mathcal{M} \vDash (\forall y)\left(\phi\left[\alpha_{(s_{i_1}^{\mathcal{M}}[\alpha])/x_{i_1},\ldots,(s_{i_k}^{\mathcal{M}}[\alpha])/x_{i_k}}\right]\right), \\
&\text{iff } \mathcal{M} \vDash (\forall y)\left(\phi\left[\alpha_{(s_1^{\mathcal{M}}[\alpha])/x_1,\ldots,(s_n^{\mathcal{M}}[\alpha])/x_n}\right]\right),
\end{aligned}$$

where we have also used that y does not occur in any of the s_i to get from the second line to the third (by Lemma 2.1.7), i.e. to get that:

$$s_{i_1}^{\mathcal{M}}[\alpha_{b/y}]) = s_{i_1}^{\mathcal{M}}[\alpha])$$

and that $x_{i_1}, \ldots, x_{i_k}$ are the only free variables of $(\forall x)\phi_1$ to get from the third line to the fourth (by Theorem 2.2.7).

- Case 2. If $y \in \bigcup_{1 \le i \le n} \mathsf{Var}(s_i)$ then:

$$(\forall y)\phi[\bar{s}/\bar{x}] = (\forall z)(\phi[s_{i_1}, \ldots s_{i_k}, z/x_{i_1}, \ldots, x_{i_k}, y])$$

In particular, we have that:

$$\begin{aligned}
\mathcal{M} \vDash ((\forall y)\phi[\bar{s}/\bar{x}])[\alpha] &\text{ iff } \mathcal{M} \vDash (\forall z)(\phi[s_{i_1}, \ldots, s_{i_k}, z/x_{i_1}, \ldots, x_{i_k}, y])[\alpha] \\
&\text{iff for all } b \in M \text{ we have } \mathcal{M} \vDash (\phi[s_{i_1}, \ldots, s_{i_k}, z/x_{i_1}, \ldots, x_{i_k}, y])[\alpha_{b/z}] \\
&\text{iff for all } b \in M \text{ we have } \mathcal{M} \vDash \phi\left[\alpha_{(s_{i_1}^{\mathcal{M}}[\alpha_{b/z}])/x_{i_1},\ldots,(s_{i_k}^{\mathcal{M}}[\alpha_{b/z}])/x_{i_k},(z^{\mathcal{M}}[\alpha_{b/z}])/y)}\right] \\
&\text{iff for all } b \in M \text{ we have } \mathcal{M} \vDash \phi\left[\alpha_{(s_1^{\mathcal{M}}[\alpha_{b/z}])/x_1,\ldots,(s_n^{\mathcal{M}}[\alpha_{b/z}])/x_n,b/y)}\right] \\
&\text{iff } \mathcal{M} \vDash (\forall y)\left(\phi\left[\alpha_{(s_{i_1}^{\mathcal{M}}[\alpha])/x_{i_1},\ldots,(s_{i_k}^{\mathcal{M}}[\alpha])/x_{i_k}}\right]\right) \\
&\text{iff } \mathcal{M} \vDash (\forall y)\left(\phi\left[\alpha_{(s_1^{\mathcal{M}}[\alpha])/x_1,\ldots,(s_n^{\mathcal{M}}[\alpha])/x_n}\right]\right),
\end{aligned}$$

where, again we've used the same facts as in the other case to go down the bi-implications, also noting that $(z^{\mathcal{M}}[\alpha_{b/z}]) = b$, by definition.

□

Exercise 3.2.8. Finish up the unwritten cases in the proof of the Substitution Lemma.

The next, and final lemma before we move on up to better and bigger things says something obvious, if we change a variable in a formula for a fresh variable and then change it back to the original variable, then we achieved nothing. The proof is left as an exercise.

Lemma 3.2.9. *If y is a variable with no occurrence in ϕ then $(\phi[y/x])[x/y] = \phi$.*

Proof. HW4 $\square$