# Chapter udf

# Semantics of First-Order Logic

## syn.1 Introduction

fol:syn:its:sec

Giving the meaning of expressions is the domain of semantics. The central concept in semantics is that of satisfaction in a structure. A structure gives meaning to the building blocks of the language: a domain is a non-empty set of objects. The quantifiers are interpreted as ranging over this domain, constant symbols are assigned elements in the domain, function symbols are assigned functions from the domain to itself, and predicate symbols are assigned relations on the domain. The domain together with assignments to the basic vocabulary constitutes a structure. Variables may appear in formulas, and in order to give a semantics, we also have to assign elements of the domain to them—this is a variable assignment. The satisfaction relation, finally, brings these together. A formula may be satisfied in a structure $\mathfrak{M}$ relative to a variable assignment $s$, written as $\mathfrak{M}, s \vDash \varphi$. This relation is also defined by induction on the structure of $\varphi$, using the truth tables for the logical connectives to define, say, satisfaction of $(\varphi \land \psi)$ in terms of satisfaction (or not) of $\varphi$ and $\psi$. It then turns out that the variable assignment is irrelevant if the formula $\varphi$ is a sentence, i.e., has no free variables, and so we can talk of sentences being simply satisfied (or not) in structures.

On the basis of the satisfaction relation $\mathfrak{M} \vDash \varphi$ for sentences we can then define the basic semantic notions of validity, entailment, and satisfiability. A sentence is valid, $\vDash \varphi$, if every structure satisfies it. It is entailed by a set of sentences, $\Gamma \vDash \varphi$, if every structure that satisfies all the sentences in $\Gamma$ also satisfies $\varphi$. And a set of sentences is satisfiable if some structure satisfies all sentences in it at the same time. Because formulas are inductively defined, and satisfaction is in turn defined by induction on the structure of formulas, we can use induction to prove properties of our semantics and to relate the semantic notions defined.

## syn.2 Structures for First-order Languages

explanation fol:syn:str: sec

First-order languages are, by themselves, *uninterpreted:* the constant symbols, function symbols, and predicate symbols have no specific meaning attached to them. Meanings are given by specifying a *structure*. It specifies the *domain*, i.e., the objects which the constant symbols pick out, the function symbols operate on, and the quantifiers range over. In addition, it specifies which constant symbols pick out which objects, how a function symbol maps objects to objects, and which objects the predicate symbols apply to. Structures are the basis for *semantic* notions in logic, e.g., the notion of consequence, validity, satisfiability. They are variously called "structures," "interpretations," or "models" in the literature.

**Definition syn.1 (Structures).** A *structure* $\mathfrak{M}$, for a language $\mathcal{L}$ of first-order logic consists of the following elements:

1. *Domain:* a non-empty set, $|\mathfrak{M}|$
2. *Interpretation of constant symbols:* for each constant symbol $c$ of $\mathcal{L}$, an element $c^{\mathfrak{M}} \in |\mathfrak{M}|$
3. *Interpretation of predicate symbols:* for each $n$-place predicate symbol $R$ of $\mathcal{L}$ (other than $=$), an $n$-place relation $R^{\mathfrak{M}} \subseteq |\mathfrak{M}|^n$
4. *Interpretation of function symbols:* for each $n$-place function symbol $f$ of $\mathcal{L}$, an $n$-place function $f^{\mathfrak{M}} \colon |\mathfrak{M}|^n \to |\mathfrak{M}|$

**Example syn.2.** A structure $\mathfrak{M}$ for the language of arithmetic consists of a set, an element of $|\mathfrak{M}|$, $\mathsf{o}^{\mathfrak{M}}$, as interpretation of the constant symbol $\mathsf{o}$, a one-place function $\prime^{\mathfrak{M}} \colon |\mathfrak{M}| \to |\mathfrak{M}|$, two two-place functions $+^{\mathfrak{M}}$ and $\times^{\mathfrak{M}}$, both $|\mathfrak{M}|^2 \to |\mathfrak{M}|$, and a two-place relation $<^{\mathfrak{M}} \subseteq |\mathfrak{M}|^2$.

An obvious example of such a structure is the following:

1. $|\mathfrak{N}| = \mathbb{N}$
2. $\mathsf{o}^{\mathfrak{N}} = 0$
3. $\prime^{\mathfrak{N}}(n) = n + 1$ for all $n \in \mathbb{N}$
4. $+^{\mathfrak{N}}(n, m) = n + m$ for all $n, m \in \mathbb{N}$
5. $\times^{\mathfrak{N}}(n, m) = n \cdot m$ for all $n, m \in \mathbb{N}$
6. $<^{\mathfrak{N}} = \{\langle n, m\rangle : n \in \mathbb{N}, m \in \mathbb{N}, n < m\}$

The structure $\mathfrak{N}$ for $\mathcal{L}_A$ so defined is called the *standard model of arithmetic*, because it interprets the non-logical constants of $\mathcal{L}_A$ exactly how you would expect.

However, there are many other possible structures for $\mathcal{L}_A$. For instance, we might take as the domain the set $\mathbb{Z}$ of integers instead of $\mathbb{N}$, and define the interpretations of $\mathsf{o}$, $\prime$, $+$, $\times$, $<$ accordingly. But we can also define structures for $\mathcal{L}_A$ which have nothing even remotely to do with numbers.

**Example syn.3.** A structure $\mathfrak{M}$ for the language $\mathcal{L}_Z$ of set theory requires just a set and a single-two place relation. So technically, e.g., the set of people plus the relation "$x$ is older than $y$" could be used as a structure for $\mathcal{L}_Z$, as well as $\mathbb{N}$ together with $n \geq m$ for $n, m \in \mathbb{N}$.

A particularly interesting structure for $\mathcal{L}_Z$ in which the elements of the domain are actually sets, and the interpretation of $\in$ actually is the relation "$x$ is an element of $y$" is the structure $\mathfrak{HF}$ of *hereditarily finite sets*:

1. $|\mathfrak{HF}| = \emptyset \cup \wp(\emptyset) \cup \wp(\wp(\emptyset)) \cup \wp(\wp(\wp(\emptyset))) \cup \dots;$

2. $\in^{\mathfrak{HF}} = \{\langle x, y\rangle : x, y \in |\mathfrak{HF}|, x \in y\}$.

The stipulations we make as to what counts as a structure impact our logic. For example, the choice to prevent empty domains ensures, given the usual account of satisfaction (or truth) for quantified sentences, that $\exists x\,(\varphi(x) \lor \lnot\varphi(x))$ is valid—that is, a logical truth. And the stipulation that all constant symbols must refer to an object in the domain ensures that the existential generalization is a sound pattern of inference: $\varphi(a)$, therefore $\exists x\,\varphi(x)$. If we allowed names to refer outside the domain, or to not refer, then we would be on our way to a *free logic*, in which existential generalization requires an additional premise: $\varphi(a)$ and $\exists x\, x = a$, therefore $\exists x\,\varphi(x)$. digression

## syn.3 Covered Structures for First-order Languages

fol:syn:cov:sec Recall that a term is *closed* if it contains no variables. explanation

**Definition syn.4 (Value of closed terms).** If $t$ is a closed term of the language $\mathcal{L}$ and $\mathfrak{M}$ is a structure for $\mathcal{L}$, the *value* $\mathrm{Val}^{\mathfrak{M}}(t)$ is defined as follows:

1. If $t$ is just the constant symbol $c$, then $\mathrm{Val}^{\mathfrak{M}}(c) = c^{\mathfrak{M}}$.

2. If $t$ is of the form $f(t_1, \dots, t_n)$, then
$$\mathrm{Val}^{\mathfrak{M}}(t) = f^{\mathfrak{M}}(\mathrm{Val}^{\mathfrak{M}}(t_1), \dots, \mathrm{Val}^{\mathfrak{M}}(t_n)).$$

**Definition syn.5 (Covered structure).** A structure is *covered* if every element of the domain is the value of some closed term.

**Example syn.6.** Let $\mathcal{L}$ be the language with constant symbols $\mathit{zero}$, $\mathit{one}$, $\mathit{two}$, …, the binary predicate symbol $<$, and the binary function symbols $+$ and $\times$. Then a structure $\mathfrak{M}$ for $\mathcal{L}$ is the one with domain $|\mathfrak{M}| = \{0, 1, 2, \dots\}$ and assignments $\mathit{zero}^{\mathfrak{M}} = 0$, $\mathit{one}^{\mathfrak{M}} = 1$, $\mathit{two}^{\mathfrak{M}} = 2$, and so forth. For the binary relation symbol $<$, the set $<^{\mathfrak{M}}$ is the set of all pairs $\langle c_1, c_2\rangle \in |\mathfrak{M}|^2$ such that $c_1$ is less than $c_2$: for example, $\langle 1, 3\rangle \in <^{\mathfrak{M}}$ but $\langle 2, 2\rangle \notin <^{\mathfrak{M}}$. For the binary function symbol $+$, define $+^{\mathfrak{M}}$ in the usual way—for example, $+^{\mathfrak{M}}(2, 3)$ maps to 5, and similarly for the binary function symbol $\times$. Hence, the value of

$\mathit{four}$ is just 4, and the value of $\times(\mathit{two}, +(\mathit{three}, \mathit{zero}))$ (or in infix notation, $\mathit{two} \times (\mathit{three} + \mathit{zero}))$ is

$$\begin{aligned}
\mathrm{Val}^{\mathfrak{M}}(\times(\mathit{two}, +(\mathit{three}, \mathit{zero}))) = \\
&= \times^{\mathfrak{M}}(\mathrm{Val}^{\mathfrak{M}}(\mathit{two}), \mathrm{Val}^{\mathfrak{M}}(+(\mathit{three}, \mathit{zero}))) \\
&= \times^{\mathfrak{M}}(\mathrm{Val}^{\mathfrak{M}}(\mathit{two}), +^{\mathfrak{M}}(\mathrm{Val}^{\mathfrak{M}}(\mathit{three}), \mathrm{Val}^{\mathfrak{M}}(\mathit{zero}))) \\
&= \times^{\mathfrak{M}}(\mathit{two}^{\mathfrak{M}}, +^{\mathfrak{M}}(\mathit{three}^{\mathfrak{M}}, \mathit{zero}^{\mathfrak{M}})) \\
&= \times^{\mathfrak{M}}(2, +^{\mathfrak{M}}(3, 0)) \\
&= \times^{\mathfrak{M}}(2, 3) \\
&= 6
\end{aligned}$$

**Problem syn.1.** Is $\mathfrak{N}$, the standard model of arithmetic, covered? Explain.

## syn.4 Satisfaction of a Formula in a Structure

explanation

fol:syn:sat:sec

The basic notion that relates expressions such as terms and formulas, on the one hand, and structures on the other, are those of *value* of a term and *satisfaction* of a formula. Informally, the value of a term is an element of a structure—if the term is just a constant, its value is the object assigned to the constant by the structure, and if it is built up using function symbols, the value is computed from the values of constants and the functions assigned to the functions in the term. A formula is *satisfied* in a structure if the interpretation given to the predicates makes the formula true in the domain of the structure. This notion of satisfaction is specified inductively: the specification of the structure directly states when atomic formulas are satisfied, and we define when a complex formula is satisfied depending on the main connective or quantifier and whether or not the immediate subformulas are satisfied.

The case of the quantifiers here is a bit tricky, as the immediate subformula of a quantified formula has a free variable, and structures don't specify the values of variables. In order to deal with this difficulty, we also introduce *variable assignments* and define satisfaction not with respect to a structure alone, but with respect to a structure plus a variable assignment.

**Definition syn.7 (Variable Assignment).** A *variable assignment* $s$ for a structure $\mathfrak{M}$ is a function which maps each variable to an element of $|\mathfrak{M}|$, i.e., $s \colon \mathrm{Var} \to |\mathfrak{M}|$.

explanation

A structure assigns a value to each constant symbol, and a variable assignment to each variable. But we want to use terms built up from them to also name elements of the domain. For this we define the value of terms inductively. For constant symbols and variables the value is just as the structure or the variable assignment specifies it; for more complex terms it is computed recursively using the functions the structure assigns to the function symbols.

**Definition syn.8 (Value of Terms).** If $t$ is a term of the language $\mathcal{L}$, $\mathfrak{M}$ is a structure for $\mathcal{L}$, and $s$ is a variable assignment for $\mathfrak{M}$, the *value* $\mathrm{Val}^{\mathfrak{M}}_s(t)$ is defined as follows:

1. $t \equiv c$: $\mathrm{Val}^{\mathfrak{M}}_s(t) = c^{\mathfrak{M}}$.
2. $t \equiv x$: $\mathrm{Val}^{\mathfrak{M}}_s(t) = s(x)$.
3. $t \equiv f(t_1, \dots, t_n)$:

$$\mathrm{Val}^{\mathfrak{M}}_s(t) = f^{\mathfrak{M}}(\mathrm{Val}^{\mathfrak{M}}_s(t_1), \dots, \mathrm{Val}^{\mathfrak{M}}_s(t_n)).$$

**Definition syn.9 ($x$-Variant).** If $s$ is a variable assignment for a structure $\mathfrak{M}$, then any variable assignment $s'$ for $\mathfrak{M}$ which differs from $s$ at most in what it assigns to $x$ is called an *$x$-variant* of $s$. If $s'$ is an $x$-variant of $s$ we write $s' \sim_x s$.

Note that an $x$-variant of an assignment $s$ does not *have* to assign something different to $x$. In fact, every assignment counts as an $x$-variant of itself. explanation

**Definition syn.10.** If $s$ is a variable assignment for a structure $\mathfrak{M}$ and $m \in |\mathfrak{M}|$, then the assignment $s[m/x]$ is the variable assignment defined by

$$s[m/x](y) = \begin{cases} m & \text{if } y \equiv x \\ s(y) & \text{otherwise.} \end{cases}$$

In other words, $s[m/x]$ is the particular $x$-variant of $s$ which assigns the domain element $m$ to $x$, and assigns the same things to variables other than $x$ that $s$ does.

fol:syn:sat: defn:satisfaction **Definition syn.11 (Satisfaction).** Satisfaction of a formula $\varphi$ in a structure $\mathfrak{M}$ relative to a variable assignment $s$, in symbols: $\mathfrak{M}, s \vDash \varphi$, is defined recursively as follows. (We write $\mathfrak{M}, s \nvDash \varphi$ to mean "not $\mathfrak{M}, s \vDash \varphi$.")

1. $\varphi \equiv \bot$: $\mathfrak{M}, s \nvDash \varphi$.
2. $\varphi \equiv \top$: $\mathfrak{M}, s \vDash \varphi$.
3. $\varphi \equiv R(t_1, \dots, t_n)$: $\mathfrak{M}, s \vDash \varphi$ iff $\langle \mathrm{Val}^{\mathfrak{M}}_s(t_1), \dots, \mathrm{Val}^{\mathfrak{M}}_s(t_n) \rangle \in R^{\mathfrak{M}}$.
4. $\varphi \equiv t_1 = t_2$: $\mathfrak{M}, s \vDash \varphi$ iff $\mathrm{Val}^{\mathfrak{M}}_s(t_1) = \mathrm{Val}^{\mathfrak{M}}_s(t_2)$.
5. $\varphi \equiv \neg\psi$: $\mathfrak{M}, s \vDash \varphi$ iff $\mathfrak{M}, s \nvDash \psi$.
6. $\varphi \equiv (\psi \land \chi)$: $\mathfrak{M}, s \vDash \varphi$ iff $\mathfrak{M}, s \vDash \psi$ and $\mathfrak{M}, s \vDash \chi$.
7. $\varphi \equiv (\psi \lor \chi)$: $\mathfrak{M}, s \vDash \varphi$ iff $\mathfrak{M}, s \vDash \psi$ or $\mathfrak{M}, s \vDash \chi$ (or both).
8. $\varphi \equiv (\psi \to \chi)$: $\mathfrak{M}, s \vDash \varphi$ iff $\mathfrak{M}, s \nvDash \psi$ or $\mathfrak{M}, s \vDash \chi$ (or both).

9. $\varphi \equiv (\psi \leftrightarrow \chi)$: $\mathfrak{M}, s \vDash \varphi$ iff either both $\mathfrak{M}, s \vDash \psi$ and $\mathfrak{M}, s \vDash \chi$, or neither $\mathfrak{M}, s \vDash \psi$ nor $\mathfrak{M}, s \vDash \chi$.

10. $\varphi \equiv \forall x\, \psi$: $\mathfrak{M}, s \vDash \varphi$ iff for every element $m \in |\mathfrak{M}|$, $\mathfrak{M}, s[m/x] \vDash \psi$.

11. $\varphi \equiv \exists x\, \psi$: $\mathfrak{M}, s \vDash \varphi$ iff for at least one element $m \in |\mathfrak{M}|$, $\mathfrak{M}, s[m/x] \vDash \psi$.

explanation

The variable assignments are important in the last two clauses. We cannot define satisfaction of $\forall x\, \psi(x)$ by "for all $m \in |\mathfrak{M}|$, $\mathfrak{M} \vDash \psi(m)$." We cannot define satisfaction of $\exists x\, \psi(x)$ by "for at least one $m \in |\mathfrak{M}|$, $\mathfrak{M} \vDash \psi(m)$." The reason is that if $m \in |\mathfrak{M}|$, it is not a symbol of the language, and so $\psi(m)$ is not a formula (that is, $\psi[m/x]$ is undefined). We also cannot assume that we have constant symbols or terms available that name every element of $\mathfrak{M}$, since there is nothing in the definition of structures that requires it. In the standard language, the set of constant symbols is denumerable, so if $|\mathfrak{M}|$ is not enumerable there aren't even enough constant symbols to name every object.

We solve this problem by introducing variable assignments, which allow us to link variables directly with elements of the domain. Then instead of saying that, e.g., $\exists x\, \psi(x)$ is satisfied in $\mathfrak{M}$ iff for at least one $m \in |\mathfrak{M}|$, we say it is satisfied in $\mathfrak{M}$ *relative to* $s$ iff $\psi(x)$ is satisfied relative to $s[m/x]$ for at least one $m \in |\mathfrak{M}|$.

**Example syn.12.** Let $\mathcal{L} = \{a, b, f, R\}$ where $a$ and $b$ are constant symbols, $f$ is a two-place function symbol, and $R$ is a two-place predicate symbol. Consider the structure $\mathfrak{M}$ defined by:

1. $|\mathfrak{M}| = \{1, 2, 3, 4\}$

2. $a^{\mathfrak{M}} = 1$

3. $b^{\mathfrak{M}} = 2$

4. $f^{\mathfrak{M}}(x, y) = x + y$ if $x + y \leq 3$ and $= 3$ otherwise.

5. $R^{\mathfrak{M}} = \{\langle 1, 1\rangle, \langle 1, 2\rangle, \langle 2, 3\rangle, \langle 2, 4\rangle\}$

The function $s(x) = 1$ that assigns $1 \in |\mathfrak{M}|$ to every variable is a variable assignment for $\mathfrak{M}$.

Then

$$\mathrm{Val}^{\mathfrak{M}}_s(f(a, b)) = f^{\mathfrak{M}}(\mathrm{Val}^{\mathfrak{M}}_s(a), \mathrm{Val}^{\mathfrak{M}}_s(b)).$$

Since $a$ and $b$ are constant symbols, $\mathrm{Val}^{\mathfrak{M}}_s(a) = a^{\mathfrak{M}} = 1$ and $\mathrm{Val}^{\mathfrak{M}}_s(b) = b^{\mathfrak{M}} = 2$. So

$$\mathrm{Val}^{\mathfrak{M}}_s(f(a, b)) = f^{\mathfrak{M}}(1, 2) = 1 + 2 = 3.$$

To compute the value of $f(f(a, b), a)$ we have to consider

$$\mathrm{Val}^{\mathfrak{M}}_s(f(f(a, b), a)) = f^{\mathfrak{M}}(\mathrm{Val}^{\mathfrak{M}}_s(f(a, b)), \mathrm{Val}^{\mathfrak{M}}_s(a)) = f^{\mathfrak{M}}(3, 1) = 3,$$

since $3+1>3$. Since $s(x)=1$ and $\mathrm{Val}^{\mathfrak{M}}_s(x)=s(x)$, we also have

$$\mathrm{Val}^{\mathfrak{M}}_s(f(f(a,b),x)) = f^{\mathfrak{M}}(\mathrm{Val}^{\mathfrak{M}}_s(f(a,b)), \mathrm{Val}^{\mathfrak{M}}_s(x)) = f^{\mathfrak{M}}(3,1) = 3,$$

An atomic formula $R(t_1,t_2)$ is satisfied if the tuple of values of its arguments, i.e., $\langle \mathrm{Val}^{\mathfrak{M}}_s(t_1), \mathrm{Val}^{\mathfrak{M}}_s(t_2)\rangle$, is an element of $R^{\mathfrak{M}}$. So, e.g., we have $\mathfrak{M}, s \vDash R(b, f(a,b))$ since $\langle \mathrm{Val}^{\mathfrak{M}}(b), \mathrm{Val}^{\mathfrak{M}}(f(a,b))\rangle = \langle 2,3\rangle \in R^{\mathfrak{M}}$, but $\mathfrak{M}, s \nvDash R(x, f(a,b))$ since $\langle 1,3\rangle \notin R^{\mathfrak{M}}[s]$.

To determine if a non-atomic formula $\varphi$ is satisfied, you apply the clauses in the inductive definition that applies to the main connective. For instance, the main connective in $R(a,a) \to (R(b,x) \lor R(x,b))$ is the $\to$, and

$$\begin{aligned} &\mathfrak{M}, s \vDash R(a,a) \to (R(b,x) \lor R(x,b)) \text{ iff} \\ &\qquad \mathfrak{M}, s \nvDash R(a,a) \text{ or } \mathfrak{M}, s \vDash R(b,x) \lor R(x,b) \end{aligned}$$

Since $\mathfrak{M}, s \vDash R(a,a)$ (because $\langle 1,1\rangle \in R^{\mathfrak{M}}$) we can't yet determine the answer and must first figure out if $\mathfrak{M}, s \vDash R(b,x) \lor R(x,b)$:

$$\begin{aligned} &\mathfrak{M}, s \vDash R(b,x) \lor R(x,b) \text{ iff} \\ &\qquad \mathfrak{M}, s \vDash R(b,x) \text{ or } \mathfrak{M}, s \vDash R(x,b) \end{aligned}$$

And this is the case, since $\mathfrak{M}, s \vDash R(x,b)$ (because $\langle 1,2\rangle \in R^{\mathfrak{M}}$).

Recall that an $x$-variant of $s$ is a variable assignment that differs from $s$ at most in what it assigns to $x$. For every element of $|\mathfrak{M}|$, there is an $x$-variant of $s$:

$$\begin{aligned} s_1 &= s[1/x], & s_2 &= s[2/x], \\ s_3 &= s[3/x], & s_4 &= s[4/x]. \end{aligned}$$

So, e.g., $s_2(x)=2$ and $s_2(y)=s(y)=1$ for all variables $y$ other than $x$. These are all the $x$-variants of $s$ for the structure $\mathfrak{M}$, since $|\mathfrak{M}| = \{1,2,3,4\}$. Note, in particular, that $s_1 = s$ ($s$ is always an $x$-variant of itself).

To determine if an existentially quantified formula $\exists x\, \varphi(x)$ is satisfied, we have to determine if $\mathfrak{M}, s[m/x] \vDash \varphi(x)$ for at least one $m \in |\mathfrak{M}|$. So,

$$\mathfrak{M}, s \vDash \exists x\, (R(b,x) \lor R(x,b)),$$

since $\mathfrak{M}, s[1/x] \vDash R(b,x) \lor R(x,b)$ ($s[3/x]$ would also fit the bill). But,

$$\mathfrak{M}, s \nvDash \exists x\, (R(b,x) \land R(x,b))$$

since, whichever $m \in |\mathfrak{M}|$ we pick, $\mathfrak{M}, s[m/x] \nvDash R(b,x) \land R(x,b)$.

To determine if a universally quantified formula $\forall x\, \varphi(x)$ is satisfied, we have to determine if $\mathfrak{M}, s[m/x] \vDash \varphi(x)$ for all $m \in |\mathfrak{M}|$. So,

$$\mathfrak{M}, s \vDash \forall x\, (R(x,a) \to R(a,x)),$$

since $\mathfrak{M}, s[m/x] \vDash R(x,a) \rightarrow R(a,x)$ for all $m \in |\mathfrak{M}|$. For $m = 1$, we have $\mathfrak{M}, s[1/x] \vDash R(a,x)$ so the consequent is true; for $m = 2$, 3, and 4, we have $\mathfrak{M}, s[m/x] \nvDash R(x,a)$, so the antecedent is false. But,

$$\mathfrak{M}, s \nvDash \forall x\, (R(a,x) \rightarrow R(x,a))$$

since $\mathfrak{M}, s[2/x] \nvDash R(a,x) \rightarrow R(x,a)$ (because $\mathfrak{M}, s[2/x] \vDash R(a,x)$ and $\mathfrak{M}, s[2/x] \nvDash R(x,a)$).

For a more complicated case, consider

$$\forall x\, (R(a,x) \rightarrow \exists y\, R(x,y)).$$

Since $\mathfrak{M}, s[3/x] \nvDash R(a,x)$ and $\mathfrak{M}, s[4/x] \nvDash R(a,x)$, the interesting cases where we have to worry about the consequent of the conditional are only $m = 1$ and $= 2$. Does $\mathfrak{M}, s[1/x] \vDash \exists y\, R(x,y)$ hold? It does if there is at least one $n \in |\mathfrak{M}|$ so that $\mathfrak{M}, s[1/x][n/y] \vDash R(x,y)$. In fact, if we take $n = 1$, we have $s[1/x][n/y] = s[1/y] = s$. Since $s(x) = 1$, $s(y) = 1$, and $\langle 1, 1 \rangle \in R^{\mathfrak{M}}$, the answer is yes.

To determine if $\mathfrak{M}, s[2/x] \vDash \exists y\, R(x,y)$, we have to look at the variable assignments $s[2/x][n/y]$. Here, for $n = 1$, this assignment is $s_2 = s[2/x]$, which does not satisfy $R(x,y)$ ($s_2(x) = 2$, $s_2(y) = 1$, and $\langle 2, 1 \rangle \notin R^{\mathfrak{M}}$). However, consider $s[2/x][3/y] = s_2[3/y]$. $\mathfrak{M}, s_2[3/y] \vDash R(x,y)$ since $\langle 2, 3 \rangle \in R^{\mathfrak{M}}$, and so $\mathfrak{M}, s_2 \vDash \exists y\, R(x,y)$.

So, for all $n \in |\mathfrak{M}|$, either $\mathfrak{M}, s[m/x] \nvDash R(a,x)$ (if $m = 3$, 4) or $\mathfrak{M}, s[m/x] \vDash \exists y\, R(x,y)$ (if $m = 1$, 2), and so

$$\mathfrak{M}, s \vDash \forall x\, (R(a,x) \rightarrow \exists y\, R(x,y)).$$

On the other hand,

$$\mathfrak{M}, s \nvDash \exists x\, (R(a,x) \land \forall y\, R(x,y)).$$

We have $\mathfrak{M}, s[m/x] \vDash R(a,x)$ only for $m = 1$ and $m = 2$. But for both of these values of $m$, there is in turn an $n \in |\mathfrak{M}|$, namely $n = 4$, so that $\mathfrak{M}, s[m/x][n/y] \nvDash R(x,y)$ and so $\mathfrak{M}, s[m/x] \nvDash \forall y\, R(x,y)$ for $m = 1$ and $m = 2$. In sum, there is no $m \in |\mathfrak{M}|$ such that $\mathfrak{M}, s[m/x] \vDash R(a,x) \land \forall y\, R(x,y)$.

**Problem syn.2.** Let $\mathcal{L} = \{c, f, A\}$ with one constant symbol, one one-place function symbol and one two-place predicate symbol, and let the structure $\mathfrak{M}$ be given by

1. $|\mathfrak{M}| = \{1, 2, 3\}$
2. $c^{\mathfrak{M}} = 3$
3. $f^{\mathfrak{M}}(1) = 2, f^{\mathfrak{M}}(2) = 3, f^{\mathfrak{M}}(3) = 2$
4. $A^{\mathfrak{M}} = \{\langle 1, 2 \rangle, \langle 2, 3 \rangle, \langle 3, 3 \rangle\}$

(a) Let $s(v) = 1$ for all variables $v$. Find out whether

$$\mathfrak{M}, s \vDash \exists x\, (A(f(z), c) \rightarrow \forall y\, (A(y, x) \lor A(f(y), x)))$$

Explain why or why not.

(b) Give a different structure and variable assignment in which the formula is not satisfied.

## syn.5 Variable Assignments

fol:syn:ass: sec

A variable assignment $s$ provides a value for *every* variable—and there are infinitely many of them. This is of course not necessary. We require variable assignments to assign values to all variables simply because it makes things a lot easier. The value of a term $t$, and whether or not a formula $\varphi$ is satisfied in a structure with respect to $s$, only depend on the assignments $s$ makes to the variables in $t$ and the free variables of $\varphi$. This is the content of the next two propositions. To make the idea of "depends on" precise, we show that any two variable assignments that agree on all the variables in $t$ give the same value, and that $\varphi$ is satisfied relative to one iff it is satisfied relative to the other if two variable assignments agree on all free variables of $\varphi$.

explanation

*fol:syn:ass: prop:valindep*

**Proposition syn.13.** *If the variables in a term $t$ are among $x_1, \dots, x_n$, and $s_1(x_i) = s_2(x_i)$ for $i = 1, \dots, n$, then $\mathrm{Val}^{\mathfrak{M}}_{s_1}(t) = \mathrm{Val}^{\mathfrak{M}}_{s_2}(t)$.*

*Proof.* By induction on the complexity of $t$. For the base case, $t$ can be a constant symbol or one of the variables $x_1, \dots, x_n$. If $t = c$, then $\mathrm{Val}^{\mathfrak{M}}_{s_1}(t) = c^{\mathfrak{M}} = \mathrm{Val}^{\mathfrak{M}}_{s_2}(t)$. If $t = x_i$, $s_1(x_i) = s_2(x_i)$ by the hypothesis of the proposition, and so $\mathrm{Val}^{\mathfrak{M}}_{s_1}(t) = s_1(x_i) = s_2(x_i) = \mathrm{Val}^{\mathfrak{M}}_{s_2}(t)$.

For the inductive step, assume that $t = f(t_1, \dots, t_k)$ and that the claim holds for $t_1, \dots, t_k$. Then

$$\begin{aligned}\mathrm{Val}^{\mathfrak{M}}_{s_1}(t) &= \mathrm{Val}^{\mathfrak{M}}_{s_1}(f(t_1, \dots, t_k)) = \\ &= f^{\mathfrak{M}}(\mathrm{Val}^{\mathfrak{M}}_{s_1}(t_1), \dots, \mathrm{Val}^{\mathfrak{M}}_{s_1}(t_k))\end{aligned}$$

For $j = 1, \dots, k$, the variables of $t_j$ are among $x_1, \dots, x_n$. By induction hypothesis, $\mathrm{Val}^{\mathfrak{M}}_{s_1}(t_j) = \mathrm{Val}^{\mathfrak{M}}_{s_2}(t_j)$. So,

$$\begin{aligned}\mathrm{Val}^{\mathfrak{M}}_{s_1}(t) &= \mathrm{Val}^{\mathfrak{M}}_{s_1}(f(t_1, \dots, t_k)) = \\ &= f^{\mathfrak{M}}(\mathrm{Val}^{\mathfrak{M}}_{s_1}(t_1), \dots, \mathrm{Val}^{\mathfrak{M}}_{s_1}(t_k)) = \\ &= f^{\mathfrak{M}}(\mathrm{Val}^{\mathfrak{M}}_{s_2}(t_1), \dots, \mathrm{Val}^{\mathfrak{M}}_{s_2}(t_k)) = \\ &= \mathrm{Val}^{\mathfrak{M}}_{s_2}(f(t_1, \dots, t_k)) = \mathrm{Val}^{\mathfrak{M}}_{s_2}(t).\end{aligned}$$

□

*fol:syn:ass: prop:satindep*

**Proposition syn.14.** *If the free variables in $\varphi$ are among $x_1, \dots, x_n$, and $s_1(x_i) = s_2(x_i)$ for $i = 1, \dots, n$, then $\mathfrak{M}, s_1 \vDash \varphi$ iff $\mathfrak{M}, s_2 \vDash \varphi$.*

*Proof.* We use induction on the complexity of $\varphi$. For the base case, where $\varphi$ is atomic, $\varphi$ can be: $\top$, $\bot$, $R(t_1, \dots, t_k)$ for a $k$-place predicate $R$ and terms $t_1$, $\dots$, $t_k$, or $t_1 = t_2$ for terms $t_1$ and $t_2$.

1. $\varphi \equiv \top$: both $\mathfrak{M}, s_1 \vDash \varphi$ and $\mathfrak{M}, s_2 \vDash \varphi$.

2. $\varphi \equiv \bot$: both $\mathfrak{M}, s_1 \nvDash \varphi$ and $\mathfrak{M}, s_2 \nvDash \varphi$.

3. $\varphi \equiv R(t_1, \dots, t_k)$: let $\mathfrak{M}, s_1 \vDash \varphi$. Then

$$\langle \mathrm{Val}^{\mathfrak{M}}_{s_1}(t_1), \dots, \mathrm{Val}^{\mathfrak{M}}_{s_1}(t_k) \rangle \in R^{\mathfrak{M}}.$$

   For $i = 1, \dots, k$, $\mathrm{Val}^{\mathfrak{M}}_{s_1}(t_i) = \mathrm{Val}^{\mathfrak{M}}_{s_2}(t_i)$ by Proposition syn.13. So we also have $\langle \mathrm{Val}^{\mathfrak{M}}_{s_2}(t_i), \dots, \mathrm{Val}^{\mathfrak{M}}_{s_2}(t_k) \rangle \in R^{\mathfrak{M}}$.

4. $\varphi \equiv t_1 = t_2$: suppose $\mathfrak{M}, s_1 \vDash \varphi$. Then $\mathrm{Val}^{\mathfrak{M}}_{s_1}(t_1) = \mathrm{Val}^{\mathfrak{M}}_{s_1}(t_2)$. So,

$$\begin{aligned} \mathrm{Val}^{\mathfrak{M}}_{s_2}(t_1) &= \mathrm{Val}^{\mathfrak{M}}_{s_1}(t_1) && \text{(by Proposition syn.13)} \\ &= \mathrm{Val}^{\mathfrak{M}}_{s_1}(t_2) && \text{(since } \mathfrak{M}, s_1 \vDash t_1 = t_2\text{)} \\ &= \mathrm{Val}^{\mathfrak{M}}_{s_2}(t_2) && \text{(by Proposition syn.13),} \end{aligned}$$

   so $\mathfrak{M}, s_2 \vDash t_1 = t_2$.

Now assume $\mathfrak{M}, s_1 \vDash \psi$ iff $\mathfrak{M}, s_2 \vDash \psi$ for all formulas $\psi$ less complex than $\varphi$. The induction step proceeds by cases determined by the main operator of $\varphi$. In each case, we only demonstrate the forward direction of the biconditional; the proof of the reverse direction is symmetrical. In all cases except those for the quantifiers, we apply the induction hypothesis to sub-formulas $\psi$ of $\varphi$. The free variables of $\psi$ are among those of $\varphi$. Thus, if $s_1$ and $s_2$ agree on the free variables of $\varphi$, they also agree on those of $\psi$, and the induction hypothesis applies to $\psi$.

1. $\varphi \equiv \lnot\psi$: if $\mathfrak{M}, s_1 \vDash \varphi$, then $\mathfrak{M}, s_1 \nvDash \psi$, so by the induction hypothesis, $\mathfrak{M}, s_2 \nvDash \psi$, hence $\mathfrak{M}, s_2 \vDash \varphi$.

2. $\varphi \equiv \psi \land \chi$: if $\mathfrak{M}, s_1 \vDash \varphi$, then $\mathfrak{M}, s_1 \vDash \psi$ and $\mathfrak{M}, s_1 \vDash \chi$, so by induction hypothesis, $\mathfrak{M}, s_2 \vDash \psi$ and $\mathfrak{M}, s_2 \vDash \chi$. Hence, $\mathfrak{M}, s_2 \vDash \varphi$.

3. $\varphi \equiv \psi \lor \chi$: if $\mathfrak{M}, s_1 \vDash \varphi$, then $\mathfrak{M}, s_1 \vDash \psi$ or $\mathfrak{M}, s_1 \vDash \chi$. By induction hypothesis, $\mathfrak{M}, s_2 \vDash \psi$ or $\mathfrak{M}, s_2 \vDash \chi$, so $\mathfrak{M}, s_2 \vDash \varphi$.

4. $\varphi \equiv \psi \rightarrow \chi$: if $\mathfrak{M}, s_1 \vDash \varphi$, then $\mathfrak{M}, s_1 \nvDash \psi$ or $\mathfrak{M}, s_1 \vDash \chi$. By the induction hypothesis, $\mathfrak{M}, s_2 \nvDash \psi$ or $\mathfrak{M}, s_2 \vDash \chi$, so $\mathfrak{M}, s_2 \vDash \varphi$.

5. $\varphi \equiv \psi \leftrightarrow \chi$: if $\mathfrak{M}, s_1 \vDash \varphi$, then either $\mathfrak{M}, s_1 \vDash \psi$ and $\mathfrak{M}, s_1 \vDash \chi$, or $\mathfrak{M}, s_1 \nvDash \psi$ and $\mathfrak{M}, s_1 \nvDash \chi$. By the induction hypothesis, either $\mathfrak{M}, s_2 \vDash \psi$ and $\mathfrak{M}, s_2 \vDash \chi$ or $\mathfrak{M}, s_2 \nvDash \psi$ and $\mathfrak{M}, s_2 \nvDash \chi$. In either case, $\mathfrak{M}, s_2 \vDash \varphi$.

6. $\varphi \equiv \exists x\, \psi$: if $\mathfrak{M}, s_1 \vDash \varphi$, there is an $m \in |\mathfrak{M}|$ so that $\mathfrak{M}, s_1[m/x] \vDash \psi$. Let $s'_1 = s_1[m/x]$ and $s'_2 = s_2[m/x]$. The free variables of $\psi$ are among $x_1$, …, $x_n$, and $x$. $s'_1(x_i) = s'_2(x_i)$, since $s'_1$ and $s'_2$ are $x$-variants of $s_1$ and $s_2$, respectively, and by hypothesis $s_1(x_i) = s_2(x_i)$. $s'_1(x) = s'_2(x) = m$ by the way we have defined $s'_1$ and $s'_2$. Then the induction hypothesis applies to $\psi$ and $s'_1$, $s'_2$, so $\mathfrak{M}, s'_2 \vDash \psi$. Hence, since $s'_2 = s_2[m/x]$, there is an $m \in |\mathfrak{M}|$ such that $\mathfrak{M}, s_2[m/x] \vDash \psi$, and so $\mathfrak{M}, s_2 \vDash \varphi$.

7. $\varphi \equiv \forall x\, \psi$: if $\mathfrak{M}, s_1 \vDash \varphi$, then for every $m \in |\mathfrak{M}|$, $\mathfrak{M}, s_1[m/x] \vDash \psi$. We want to show that also, for every $m \in |\mathfrak{M}|$, $\mathfrak{M}, s_2[m/x] \vDash \psi$. So let $m \in |\mathfrak{M}|$ be arbitrary, and consider $s'_1 = s[m/x]$ and $s'_2 = s[m/x]$. We have that $\mathfrak{M}, s'_1 \vDash \psi$. The free variables of $\psi$ are among $x_1$, …, $x_n$, and $x$. $s'_1(x_i) = s'_2(x_i)$, since $s'_1$ and $s'_2$ are $x$-variants of $s_1$ and $s_2$, respectively, and by hypothesis $s_1(x_i) = s_2(x_i)$. $s'_1(x) = s'_2(x) = m$ by the way we have defined $s'_1$ and $s'_2$. Then the induction hypothesis applies to $\psi$ and $s'_1$, $s'_2$, and we have $\mathfrak{M}, s'_2 \vDash \psi$. This applies to every $m \in |\mathfrak{M}|$, i.e., $\mathfrak{M}, s_2[m/x] \vDash \psi$ for all $m \in |\mathfrak{M}|$, so $\mathfrak{M}, s_2 \vDash \varphi$.

By induction, we get that $\mathfrak{M}, s_1 \vDash \varphi$ iff $\mathfrak{M}, s_2 \vDash \varphi$ whenever the free variables in $\varphi$ are among $x_1$, …, $x_n$ and $s_1(x_i) = s_2(x_i)$ for $i = 1$, …, $n$. □

**Problem syn.3.** Complete the proof of Proposition syn.14.

Sentences have no free variables, so any two variable assignments assign the same things to all the (zero) free variables of any sentence. The proposition just proved then means that whether or not a sentence is satisfied in a structure relative to a variable assignment is completely independent of the assignment. We'll record this fact. It justifies the definition of satisfaction of a sentence in a structure (without mentioning a variable assignment) that follows. explanation

fol:syn:ass: cor:sat-sentence **Corollary syn.15.** *If $\varphi$ is a sentence and $s$ a variable assignment, then $\mathfrak{M}, s \vDash \varphi$ iff $\mathfrak{M}, s' \vDash \varphi$ for every variable assignment $s'$.*

*Proof.* Let $s'$ be any variable assignment. Since $\varphi$ is a sentence, it has no free variables, and so every variable assignment $s'$ trivially assigns the same things to all free variables of $\varphi$ as does $s$. So the condition of Proposition syn.14 is satisfied, and we have $\mathfrak{M}, s \vDash \varphi$ iff $\mathfrak{M}, s' \vDash \varphi$. □

fol:syn:ass: defn:satisfaction **Definition syn.16.** If $\varphi$ is a sentence, we say that a structure $\mathfrak{M}$ *satisfies* $\varphi$, $\mathfrak{M} \vDash \varphi$, iff $\mathfrak{M}, s \vDash \varphi$ for all variable assignments $s$.

If $\mathfrak{M} \vDash \varphi$, we also simply say that $\varphi$ *is true in* $\mathfrak{M}$.

fol:syn:ass: prop:sentence-sat-true **Proposition syn.17.** *Let $\mathfrak{M}$ be a structure, $\varphi$ be a sentence, and $s$ a variable assignment. $\mathfrak{M} \vDash \varphi$ iff $\mathfrak{M}, s \vDash \varphi$.*

*Proof.* Exercise. □

**Problem syn.4.** Prove Proposition syn.17

**Proposition syn.18.** *Suppose $\varphi(x)$ only contains $x$ free, and $\mathfrak{M}$ is a structure. Then:* fol:syn:ass: prop:sat-quant

1. *$\mathfrak{M} \vDash \exists x\, \varphi(x)$ iff $\mathfrak{M}, s \vDash \varphi(x)$ for at least one variable assignment $s$.*
2. *$\mathfrak{M} \vDash \forall x\, \varphi(x)$ iff $\mathfrak{M}, s \vDash \varphi(x)$ for all variable assignments $s$.*

*Proof.* Exercise. □

**Problem syn.5.** Prove Proposition syn.18.

**Problem syn.6.** Suppose $\mathcal{L}$ is a language without function symbols. Given a structure $\mathfrak{M}$, $c$ a constant symbol and $a \in |\mathfrak{M}|$, define $\mathfrak{M}[a/c]$ to be the structure that is just like $\mathfrak{M}$, except that $c^{\mathfrak{M}[a/c]} = a$. Define $\mathfrak{M} \mathrel{|\!\vDash} \varphi$ for sentences $\varphi$ by:

1. $\varphi \equiv \bot$: not $\mathfrak{M} \mathrel{|\!\vDash} \varphi$.
2. $\varphi \equiv \top$: $\mathfrak{M} \mathrel{|\!\vDash} \varphi$.
3. $\varphi \equiv R(d_1, \dots, d_n)$: $\mathfrak{M} \mathrel{|\!\vDash} \varphi$ iff $\langle d_1^{\mathfrak{M}}, \dots, d_n^{\mathfrak{M}} \rangle \in R^{\mathfrak{M}}$.
4. $\varphi \equiv d_1 = d_2$: $\mathfrak{M} \mathrel{|\!\vDash} \varphi$ iff $d_1^{\mathfrak{M}} = d_2^{\mathfrak{M}}$.
5. $\varphi \equiv \neg\psi$: $\mathfrak{M} \mathrel{|\!\vDash} \varphi$ iff not $\mathfrak{M} \mathrel{|\!\vDash} \psi$.
6. $\varphi \equiv (\psi \wedge \chi)$: $\mathfrak{M} \mathrel{|\!\vDash} \varphi$ iff $\mathfrak{M} \mathrel{|\!\vDash} \psi$ and $\mathfrak{M} \mathrel{|\!\vDash} \chi$.
7. $\varphi \equiv (\psi \vee \chi)$: $\mathfrak{M} \mathrel{|\!\vDash} \varphi$ iff $\mathfrak{M} \mathrel{|\!\vDash} \psi$ or $\mathfrak{M} \mathrel{|\!\vDash} \chi$ (or both).
8. $\varphi \equiv (\psi \rightarrow \chi)$: $\mathfrak{M} \mathrel{|\!\vDash} \varphi$ iff not $\mathfrak{M} \mathrel{|\!\vDash} \psi$ or $\mathfrak{M} \mathrel{|\!\vDash} \chi$ (or both).
9. $\varphi \equiv (\psi \leftrightarrow \chi)$: $\mathfrak{M} \mathrel{|\!\vDash} \varphi$ iff either both $\mathfrak{M} \mathrel{|\!\vDash} \psi$ and $\mathfrak{M} \mathrel{|\!\vDash} \chi$, or neither $\mathfrak{M} \mathrel{|\!\vDash} \psi$ nor $\mathfrak{M} \mathrel{|\!\vDash} \chi$.
10. $\varphi \equiv \forall x\, \psi$: $\mathfrak{M} \mathrel{|\!\vDash} \varphi$ iff for all $a \in |\mathfrak{M}|$, $\mathfrak{M}[a/c] \mathrel{|\!\vDash} \psi[c/x]$, if $c$ does not occur in $\psi$.
11. $\varphi \equiv \exists x\, \psi$: $\mathfrak{M} \mathrel{|\!\vDash} \varphi$ iff there is an $a \in |\mathfrak{M}|$ such that $\mathfrak{M}[a/c] \mathrel{|\!\vDash} \psi[c/x]$, if $c$ does not occur in $\psi$.

Let $x_1$, ..., $x_n$ be all free variables in $\varphi$, $c_1$, ..., $c_n$ constant symbols not in $\varphi$, $a_1$, ..., $a_n \in |\mathfrak{M}|$, and $s(x_i) = a_i$.

Show that $\mathfrak{M}, s \vDash \varphi$ iff $\mathfrak{M}[a_1/c_1, \dots, a_n/c_n] \mathrel{|\!\vDash} \varphi[c_1/x_1] \dots [c_n/x_n]$.

(This problem shows that it is possible to give a semantics for first-order logic that makes do without variable assignments.)

**Problem syn.7.** Suppose that $f$ is a function symbol not in $\varphi(x, y)$. Show that there is a structure $\mathfrak{M}$ such that $\mathfrak{M} \vDash \forall x\, \exists y\, \varphi(x, y)$ iff there is an $\mathfrak{M}'$ such that $\mathfrak{M}' \vDash \forall x\, \varphi(x, f(x))$.

(This problem is a special case of what's known as Skolem's Theorem; $\forall x\, \varphi(x, f(x))$ is called a *Skolem normal form* of $\forall x\, \exists y\, \varphi(x, y)$.)

## syn.6 Extensionality

fol:syn:ext:sec

explanation

Extensionality, sometimes called relevance, can be expressed informally as follows: the only factors that bear upon the satisfaction of formula $\varphi$ in a structure $\mathfrak{M}$ relative to a variable assignment $s$, are the size of the domain and the assignments made by $\mathfrak{M}$ and $s$ to the elements of the language that actually appear in $\varphi$.

One immediate consequence of extensionality is that where two structures $\mathfrak{M}$ and $\mathfrak{M}'$ agree on all the elements of the language appearing in a sentence $\varphi$ and have the same domain, $\mathfrak{M}$ and $\mathfrak{M}'$ must also agree on whether or not $\varphi$ itself is true.

*fol:syn:ext:prop:extensionality*

**Proposition syn.19 (Extensionality).** *Let $\varphi$ be a formula, and $\mathfrak{M}_1$ and $\mathfrak{M}_2$ be structures with $|\mathfrak{M}_1| = |\mathfrak{M}_2|$, and $s$ a variable assignment on $|\mathfrak{M}_1| = |\mathfrak{M}_2|$. If $c^{\mathfrak{M}_1} = c^{\mathfrak{M}_2}$, $R^{\mathfrak{M}_1} = R^{\mathfrak{M}_2}$, and $f^{\mathfrak{M}_1} = f^{\mathfrak{M}_2}$ for every constant symbol $c$, relation symbol $R$, and function symbol $f$ occurring in $\varphi$, then $\mathfrak{M}_1, s \vDash \varphi$ iff $\mathfrak{M}_2, s \vDash \varphi$.*

*Proof.* First prove (by induction on $t$) that for every term, $\mathrm{Val}^{\mathfrak{M}_1}_s(t) = \mathrm{Val}^{\mathfrak{M}_2}_s(t)$. Then prove the proposition by induction on $\varphi$, making use of the claim just proved for the induction basis (where $\varphi$ is atomic). □

**Problem syn.8.** Carry out the proof of Proposition syn.19 in detail.

*fol:syn:ext:cor:extensionality-sent*

**Corollary syn.20 (Extensionality for Sentences).** *Let $\varphi$ be a sentence and $\mathfrak{M}_1$, $\mathfrak{M}_2$ as in Proposition syn.19. Then $\mathfrak{M}_1 \vDash \varphi$ iff $\mathfrak{M}_2 \vDash \varphi$.*

*Proof.* Follows from Proposition syn.19 by Corollary syn.15. □

Moreover, the value of a term, and whether or not a structure satisfies a formula, only depend on the values of its subterms.

*fol:syn:ext:prop:ext-terms*

**Proposition syn.21.** *Let $\mathfrak{M}$ be a structure, $t$ and $t'$ terms, and $s$ a variable assignment. Then $\mathrm{Val}^{\mathfrak{M}}_s(t[t'/x]) = \mathrm{Val}^{\mathfrak{M}}_{s[\mathrm{Val}^{\mathfrak{M}}_s(t')/x]}(t)$.*

*Proof.* By induction on $t$.

1. If $t$ is a constant, say, $t \equiv c$, then $t[t'/x] = c$, and $\mathrm{Val}^{\mathfrak{M}}_s(c) = c^{\mathfrak{M}} = \mathrm{Val}^{\mathfrak{M}}_{s[\mathrm{Val}^{\mathfrak{M}}_s(t')/x]}(c)$.

2. If $t$ is a variable other than $x$, say, $t \equiv y$, then $t[t'/x] = y$, and $\mathrm{Val}^{\mathfrak{M}}_s(y) = \mathrm{Val}^{\mathfrak{M}}_{s[\mathrm{Val}^{\mathfrak{M}}_s(t')/x]}(y)$ since $s \sim_x s[\mathrm{Val}^{\mathfrak{M}}_s(t')/x]$.

3. If $t \equiv x$, then $t[t'/x] = t'$. But $\mathrm{Val}^{\mathfrak{M}}_{s[\mathrm{Val}^{\mathfrak{M}}_s(t')/x]}(x) = \mathrm{Val}^{\mathfrak{M}}_s(t')$ by definition of $s[\mathrm{Val}^{\mathfrak{M}}_s(t')/x]$.

4. If $t \equiv f(t_1, \dots, t_n)$ then we have:

$$\begin{aligned}
&\mathrm{Val}_s^{\mathfrak{M}}(t[t'/x]) = \\
&\qquad = \mathrm{Val}_s^{\mathfrak{M}}(f(t_1[t'/x], \dots, t_n[t'/x])) \\
&\qquad\qquad \text{by definition of } t[t'/x] \\
&\qquad = f^{\mathfrak{M}}(\mathrm{Val}_s^{\mathfrak{M}}(t_1[t'/x]), \dots, \mathrm{Val}_s^{\mathfrak{M}}(t_n[t'/x])) \\
&\qquad\qquad \text{by definition of } \mathrm{Val}_s^{\mathfrak{M}}(f(\dots)) \\
&\qquad = f^{\mathfrak{M}}(\mathrm{Val}^{\mathfrak{M}}_{s[\mathrm{Val}_s^{\mathfrak{M}}(t')/x]}(t_1), \dots, \mathrm{Val}^{\mathfrak{M}}_{s[\mathrm{Val}_s^{\mathfrak{M}}(t')/x]}(t_n)) \\
&\qquad\qquad \text{by induction hypothesis} \\
&\qquad = \mathrm{Val}^{\mathfrak{M}}_{s[\mathrm{Val}_s^{\mathfrak{M}}(t')/x]}(t) \text{ by definition of } \mathrm{Val}^{\mathfrak{M}}_{s[\mathrm{Val}_s^{\mathfrak{M}}(t')/x]}(f(\dots)) \qquad \square
\end{aligned}$$

**Proposition syn.22.** *Let $\mathfrak{M}$ be a structure, $\varphi$ a formula, $t'$ a term, and $s$ a variable assignment. Then $\mathfrak{M}, s \vDash \varphi[t'/x]$ iff $\mathfrak{M}, s[\mathrm{Val}_s^{\mathfrak{M}}(t')/x] \vDash \varphi$.* fol:syn:ext: prop:ext-formulas

*Proof.* Exercise. □

**Problem syn.9.** Prove Proposition syn.22

explanation

The point of Propositions syn.21 and syn.22 is the following. Suppose we have a term $t$ or a formula $\varphi$ and some term $t'$, and we want to know the value of $t[t'/x]$ or whether or not $\varphi[t'/x]$ is satisfied in a structure $\mathfrak{M}$ relative to a variable assignment $s$. Then we can either perform the substitution first and then consider the value or satisfaction relative to $\mathfrak{M}$ and $s$, or we can first determine the value $m = \mathrm{Val}_s^{\mathfrak{M}}(t')$ of $t'$ in $\mathfrak{M}$ relative to $s$, change the variable assignment to $s[m/x]$ and then consider the value of $t$ in $\mathfrak{M}$ and $s[m/x]$, or whether $\mathfrak{M}, s[m/x] \vDash \varphi$. Propositions syn.21 and syn.22 guarantee that the answer will be the same, whichever way we do it.

## syn.7 Semantic Notions

explanation fol:syn:sem: sec

Given the definition of structures for first-order languages, we can define some basic semantic properties of and relationships between sentences. The simplest of these is the notion of *validity* of a sentence. A sentence is valid if it is satisfied in every structure. Valid sentences are those that are satisfied regardless of how the non-logical symbols in it are interpreted. Valid sentences are therefore also called *logical truths*—they are true, i.e., satisfied, in any structure and hence their truth depends only on the logical symbols occurring in them and their syntactic structure, but not on the non-logical symbols or their interpretation.

**Definition syn.23 (Validity).** A sentence $\varphi$ is *valid*, $\vDash \varphi$, iff $\mathfrak{M} \vDash \varphi$ for every structure $\mathfrak{M}$.

**Definition syn.24 (Entailment).** A set of sentences $\Gamma$ *entails* a sentence $\varphi$, $\Gamma \vDash \varphi$, iff for every structure $\mathfrak{M}$ with $\mathfrak{M} \vDash \Gamma$, $\mathfrak{M} \vDash \varphi$.

**Definition syn.25 (Satisfiability).** A set of sentences $\Gamma$ is *satisfiable* if $\mathfrak{M} \vDash \Gamma$ for some structure $\mathfrak{M}$. If $\Gamma$ is not satisfiable it is called *unsatisfiable.*

**Proposition syn.26.** *A sentence $\varphi$ is valid iff $\Gamma \vDash \varphi$ for every set of sentences $\Gamma$.*

*Proof.* For the forward direction, let $\varphi$ be valid, and let $\Gamma$ be a set of sentences. Let $\mathfrak{M}$ be a structure so that $\mathfrak{M} \vDash \Gamma$. Since $\varphi$ is valid, $\mathfrak{M} \vDash \varphi$, hence $\Gamma \vDash \varphi$.

For the contrapositive of the reverse direction, let $\varphi$ be invalid, so there is a structure $\mathfrak{M}$ with $\mathfrak{M} \nvDash \varphi$. When $\Gamma = \{\top\}$, since $\top$ is valid, $\mathfrak{M} \vDash \Gamma$. Hence, there is a structure $\mathfrak{M}$ so that $\mathfrak{M} \vDash \Gamma$ but $\mathfrak{M} \nvDash \varphi$, hence $\Gamma$ does not entail $\varphi$. □

*fol:syn:sem: prop:entails-unsat*

**Proposition syn.27.** *$\Gamma \vDash \varphi$ iff $\Gamma \cup \{\neg\varphi\}$ is unsatisfiable.*

*Proof.* For the forward direction, suppose $\Gamma \vDash \varphi$ and suppose to the contrary that there is a structure $\mathfrak{M}$ so that $\mathfrak{M} \vDash \Gamma \cup \{\neg\varphi\}$. Since $\mathfrak{M} \vDash \Gamma$ and $\Gamma \vDash \varphi$, $\mathfrak{M} \vDash \varphi$. Also, since $\mathfrak{M} \vDash \Gamma \cup \{\neg\varphi\}$, $\mathfrak{M} \vDash \neg\varphi$, so we have both $\mathfrak{M} \vDash \varphi$ and $\mathfrak{M} \nvDash \varphi$, a contradiction. Hence, there can be no such structure $\mathfrak{M}$, so $\Gamma \cup \{\neg\varphi\}$ is unsatisfiable.

For the reverse direction, suppose $\Gamma \cup \{\neg\varphi\}$ is unsatisfiable. So for every structure $\mathfrak{M}$, either $\mathfrak{M} \nvDash \Gamma$ or $\mathfrak{M} \vDash \varphi$. Hence, for every structure $\mathfrak{M}$ with $\mathfrak{M} \vDash \Gamma$, $\mathfrak{M} \vDash \varphi$, so $\Gamma \vDash \varphi$. □

**Problem syn.10.**

1. Show that $\Gamma \vDash \bot$ iff $\Gamma$ is unsatisfiable.
2. Show that $\Gamma \cup \{\varphi\} \vDash \bot$ iff $\Gamma \vDash \neg\varphi$.
3. Suppose $c$ does not occur in $\varphi$ or $\Gamma$. Show that $\Gamma \vDash \forall x\, \varphi$ iff $\Gamma \vDash \varphi[c/x]$.

**Proposition syn.28.** *If $\Gamma \subseteq \Gamma'$ and $\Gamma \vDash \varphi$, then $\Gamma' \vDash \varphi$.*

*Proof.* Suppose that $\Gamma \subseteq \Gamma'$ and $\Gamma \vDash \varphi$. Let $\mathfrak{M}$ be a structure such that $\mathfrak{M} \vDash \Gamma'$; then $\mathfrak{M} \vDash \Gamma$, and since $\Gamma \vDash \varphi$, we get that $\mathfrak{M} \vDash \varphi$. Hence, whenever $\mathfrak{M} \vDash \Gamma'$, $\mathfrak{M} \vDash \varphi$, so $\Gamma' \vDash \varphi$. □

*fol:syn:sem: thm:sem-deduction*

**Theorem syn.29 (Semantic Deduction Theorem).** *$\Gamma \cup \{\varphi\} \vDash \psi$ iff $\Gamma \vDash \varphi \rightarrow \psi$.*

*Proof.* For the forward direction, let $\Gamma \cup \{\varphi\} \vDash \psi$ and let $\mathfrak{M}$ be a structure so that $\mathfrak{M} \vDash \Gamma$. If $\mathfrak{M} \vDash \varphi$, then $\mathfrak{M} \vDash \Gamma \cup \{\varphi\}$, so since $\Gamma \cup \{\varphi\}$ entails $\psi$, we get $\mathfrak{M} \vDash \psi$. Therefore, $\mathfrak{M} \vDash \varphi \rightarrow \psi$, so $\Gamma \vDash \varphi \rightarrow \psi$.

For the reverse direction, let $\Gamma \vDash \varphi \rightarrow \psi$ and $\mathfrak{M}$ be a structure so that $\mathfrak{M} \vDash \Gamma \cup \{\varphi\}$. Then $\mathfrak{M} \vDash \Gamma$, so $\mathfrak{M} \vDash \varphi \rightarrow \psi$, and since $\mathfrak{M} \vDash \varphi$, $\mathfrak{M} \vDash \psi$. Hence, whenever $\mathfrak{M} \vDash \Gamma \cup \{\varphi\}$, $\mathfrak{M} \vDash \psi$, so $\Gamma \cup \{\varphi\} \vDash \psi$. □

**Proposition syn.30.** *Let $\mathfrak{M}$ be a structure, and $\varphi(x)$ a formula with one free variable $x$, and $t$ a closed term. Then:*

fol:syn:sem:
prop:quant-terms

*1.* $\varphi(t) \vDash \exists x\, \varphi(x)$

*2.* $\forall x\, \varphi(x) \vDash \varphi(t)$

*Proof.*

1. Suppose $\mathfrak{M} \vDash \varphi(t)$. Let $s$ be a variable assignment with $s(x) = \mathrm{Val}^{\mathfrak{M}}(t)$. Then $\mathfrak{M}, s \vDash \varphi(t)$ since $\varphi(t)$ is a sentence. By Proposition syn.22, $\mathfrak{M}, s \vDash \varphi(x)$. By Proposition syn.18, $\mathfrak{M} \vDash \exists x\, \varphi(x)$.

2. Suppose $\mathfrak{M} \vDash \forall x\, \varphi(x)$. Let $s$ be a variable assignment with $s(x) = \mathrm{Val}^{\mathfrak{M}}(t)$. By Proposition syn.18, $\mathfrak{M}, s \vDash \varphi(x)$. By Proposition syn.22, $\mathfrak{M}, s \vDash \varphi(t)$. By Proposition syn.17, $\mathfrak{M} \vDash \varphi(t)$ since $\varphi(t)$ is a sentence. □

**Problem syn.11.** Complete the proof of Proposition syn.30.

# Photo Credits

# Bibliography