Notes on first-order logic

G Conant

October 8, 2020

By now in your mathematical education, you have studied (or at least heard of) many areas of mathematics which focus on the "theory" of a certain kind of abstract mathematical structure. For example: group theory, ring theory, field theory, or graph theory. These notes will introduce you to first-order logic, which provides a formal unifying framework, with which one can study any of these examples (and more). Let us recall some common mathematical structures.

Example 0.1.

1. A *group* is a tuple $(G, *, e)$ where
   - G is a set,
   - $*$ is a binary function on G,
   - e is an element of G,
   - certain axioms are satisfied.
2. An *ordered ring* is a tuple $(R, +, -, \cdot, <, 0, 1)$ where
   - R is a set,
   - $+$, $-$, $\cdot$ are binary functions on R,
   - $0, 1$ are elements of R,
   - $<$ is a binary relation on R (i.e. a subset of $R \times R$),
   - certain axioms are satisfied.
3. A *graph* is a tuple (V, E) where
   - V is a nonempty set,
   - E is a binary relation on V,
   - certain axioms are satisfied.

These notes are a rough draft; feel free to email me with corrections.

1 Languages and Structures

Definition 1.1. A **language** is a set of symbols, which are divided into three kinds: function symbols, relation symbols, and constant symbols. Formally we can write a language as

$$\mathcal{L} = \mathcal{F} \cup \mathcal{R} \cup \mathcal{C}$$

where $\mathcal{F}$, $\mathcal{R}$, and $\mathcal{C}$ are pairwise disjoint sets of symbols. Each symbol in $\mathcal{F}$ is called a **function symbol**; each symbol in $\mathcal{R}$ is called a **relation symbol**, and each symbol in $\mathcal{C}$ is called a **constant symbol**. Moreover, the function symbols and relation symbols are each given an *arity*, which is a positive integer. The notion of a language comprises all of this data.

In practice, we will refer to languages using the letter $\mathcal{L}$. We will not often use the letters $\mathcal{F}$, $\mathcal{R}$, or $\mathcal{C}$, but rather just say "f is a function symbol in $\mathcal{L}$", etc.

Example 1.2. Here are some special languages.

1. Let $\mathcal{L}_g = \{*, e\}$ be the *language of groups*, where $*$ is a binary function symbol and e is a constant symbol.

2. Let $\mathcal{L}_r = \{+, -, \cdot, 0, 1\}$ be the *language of rings (with unity)*, where $+, -, \cdot$ are binary function symbols and $0, 1$ are constant symbols.

3. Let $\mathcal{L}_o = \{<\}$ be the *language of orders*, where $<$ is a binary relation symbol. Define the *language of ordered groups* $\mathcal{L}_{og} = \mathcal{L}_g \cup \{<\}$ and the *language of ordered rings* $\mathcal{L}_{or} = \mathcal{L}_r \cup \{<\}$.

4. Let $\mathcal{L}_{gr} = \{E\}$ be the *language of graphs*, where E is a binary relation symbol.

Note that there is no substantive difference between $\mathcal{L}_o$ and $\mathcal{L}_{gr}$. Both are languages consisting of a single binary relation symbol. But we use different symbols to emphasize different things, that will only be given meaning later on (e.g., $<$ is meant to represent an order, while E is meant to represent a graph relation). Another important note is that in order to fully define a language, each symbol must be classified as a function symbol, relation symbol, or constant symbol, and each function symbol and relation symbol must be given an arity.

Definition 1.3. Let $\mathcal{L}$ be a language. An **$\mathcal{L}$-structure** $\mathcal{M}$ is given by the following data:

(i) a nonempty set M, called the **universe of** $\mathcal{M}$,

(ii) for each n-ary function symbol f in $\mathcal{L}$, a function $f^{\mathcal{M}} : M^n \to M$,

(iii) for each n-ary relation symbol R in $\mathcal{L}$, a subset $R^{\mathcal{M}} \subseteq M^n$ (i.e., an n-ary relation on M), and

(iv) for each constant symbol c in $\mathcal{L}$, an element $c^{\mathcal{M}}$ of M.

In the above definition, the function $f^{\mathcal{M}}$ is called the **interpretation of f in $\mathcal{M}$** (and similarly for relations $R^{\mathcal{M}}$ and constants $c^{\mathcal{M}}$). Note that we refer to f as a *function symbol*, while $f^{\mathcal{M}}$ is an actual function (and similarly for relation symbols vs relations, and constant symbols vs constants). One should be careful to distinguish the symbols in a language from their concrete interpretations in a particular structure.

Example 1.4. 1. Consider the language of groups $\mathcal{L}_g$. Then we can define an $\mathcal{L}_g$-structure $\mathcal{M}$ where M is $\mathbb{Z}$, $*^{\mathcal{M}}$ is $+$, and $e^{\mathcal{M}}$ is 0.

2. Consider the language of orders $\mathcal{L}_o$. Then we can define an $\mathcal{L}_g$-structure $\mathcal{M}$ where M is $\mathbb{Q}$ and $<^{\mathcal{M}}$ is the usual ordering on the rationals.

The previous examples might be somewhat misleading. In particular, the $\mathcal{L}_g$-structure we defined happened to be a group, and the $\mathcal{L}_o$-structure happened to be a (linear) order. But no part of the definition of structures implies that they have to have such nice properties. For example, we may define an $\mathcal{L}_g$-structure $\mathcal{M}$ where M is $\mathbb{N}$, $e^{\mathcal{M}}$ is 472, and $x *^{\mathcal{M}} y = x^y + \lfloor \log(x + y + 1) \rfloor$.

2 Formulas

Our next task is to define a formal syntax for expressing properties of $\mathcal{L}$-structures using the symbols in $\mathcal{L}$. To motivate the definitions, we make the following observations.

Example 2.1. Consider the $\mathcal{L}_{or}$-structure $(\mathbb{R}, +, \cdot, <, 0, 1)$. There are many more functions and relations, which are not in $\mathcal{L}_{or}$, but are still expressible using the symbols in $\mathcal{L}_{or}$. For example:

1. the *unary* function $f : \mathbb{R} \longrightarrow \mathbb{R}$ such that $f(x) = x + 1$;
2. the *ternary* relation $R = \{(x, y, z) \in \mathbb{R}^3 : x < y + z\}$.

To address this issue, we formally define how to build new functions and relations from the symbols in a given language. In particular, we define $\mathcal{L}$-*terms* and $\mathcal{L}$-*formulas*, which will be certain special strings of symbols built from:

- the symbols in $\mathcal{L}$,
- the equality sign $=$ (to be interpreted as equality),
- countably many variable symbols: e.g. u, v, w, x, y, z, or v_i for $i \in \mathbb{N}$, etc...
- the Boolean connectives $\wedge$ and $\neg$ (to be interpreted as "and" and "not", respectively),
- the universal quantifier symbol $\forall$ (to be interpreted as "for all"),
- parentheses and commas (for parsing and listing).

We will later observe that several other "natural" logical operators are expressible using these symbols (see Remark 3.2).

2.1 Terms (new functions)

Definition 2.2. Let $\mathcal{L}$ be a language. The set of **$\mathcal{L}$-terms** is the smallest set $\mathcal{T}$ satisfying the following properties:

(*i*) $c \in \mathcal{T}$ for any constant symbol c in $\mathcal{L}$,

(*ii*) $v \in \mathcal{T}$ for each variable symbol v,

(*iii*) if f is an n-ary function symbol in $\mathcal{L}$, and $t_1, \ldots, t_n \in \mathcal{T}$, then $f(t_1, \ldots, t_n) \in \mathcal{T}$.

Returning to Example 2.1, we can now express the function $f(x) = x + 1$ as an $\mathcal{L}_{or}$-term. If we pedantically follow the full formality of the definition, then this term would be:

$$+(x, 1).$$

For the sake of better comprehension, we abuse notation and write this term as $x + 1$.

As suggested by Example 2.1, we will interpret $\mathcal{L}$-terms as functions on $\mathcal{L}$-structures. To do this, it is convenient to think of constant symbols as "function symbols of arity 0". We use the convention $M^0 = \{\emptyset\}$. Given a language $\mathcal{L}$, an $\mathcal{L}$-structure $\mathcal{M}$, and a constant symbol c in $\mathcal{L}$, we identify the interpretation $c^{\mathcal{M}}$ with the 0-ary function $\emptyset \mapsto c^{\mathcal{M}}$ from M^0 to M.

Definition 2.3. Fix a language $\mathcal{L}$. Let t be an $\mathcal{L}$-term and let $\mathcal{M}$ be an $\mathcal{L}$-structure. By induction on the construction of terms, we define a function $t^{\mathcal{M}} : M^n \longrightarrow M$, where n is the number of distinct variable symbols appearing in t.

(i) If t is a constant symbol c, then $t^{\mathcal{M}} : M^0 \longrightarrow M$ such that $t^{\mathcal{M}}(\emptyset) = c^{\mathcal{M}}$.

(ii) If t is a variable symbol, then $t^{\mathcal{M}} : M \longrightarrow M$ is the identity function.

(iii) Suppose f is an m-ary function symbol, and t is the $\mathcal{L}$-term $f(t_1, \ldots, t_m)$, where $t_1, \ldots, t_m$ are $\mathcal{L}$-terms using variables from among $v_1, \ldots, v_n$. Define $t^{\mathcal{M}} : M^n \longrightarrow M$ such that

$$t^{\mathcal{M}}(\bar{a}) = f^{\mathcal{M}}(t_1^{\mathcal{M}}(\bar{a}), \ldots, t_m^{\mathcal{M}}(\bar{a})),$$

where, for $1 \leq i \leq m$, $t_i^{\mathcal{M}}(\bar{a})$ denotes the function $t_i^{\mathcal{M}}$ evaluated on the subtuple of $\bar{a}$ corresponding to the variables used in t_i (which can be $\emptyset$ if t_i is a constant symbol).

2.2 Formulas (new relations)

Definition 2.4. Let $\mathcal{L}$ be a language.

1. An **atomic $\mathcal{L}$-formula** is a string φ of symbols of one of the following forms:
   - *(i)* $(t_1 = t_2)$, where t_1, t_2 are $\mathcal{L}$-terms, or
   - *(ii)* $R(t_1, \ldots, t_n)$, where R is an n-ary relation symbol in $\mathcal{L}$ and $t_1, \ldots, t_n$ are $\mathcal{L}$-terms.
2. The set of **$\mathcal{L}$-formulas** is the smallest set $\mathcal{F}$ satisfying the following properties:
   - *(i)* any atomic $\mathcal{L}$-formula is in $\mathcal{F}$,
   - *(ii)* if $\varphi \in \mathcal{F}$ then $\neg\varphi \in \mathcal{F}$,
   - *(iii)* if $\varphi, \psi \in \mathcal{F}$ then $(\varphi \wedge \psi) \in \mathcal{F}$,
   - *(iv)* if $\varphi \in \mathcal{F}$ and v is a variable symbol, then $\forall v(\varphi) \in \mathcal{F}$.

Returning to Example 2.1, we can express the relation R as the atomic $\mathcal{L}_{or}$-formula

$$<(x, +(y, z)).$$

Once again, for the sake of comprehension and readability, we instead write: $x < y + z$.

Definition 2.5. Given $\mathcal{L}$-formula φ, and a variable v used in φ, we say v **occurs freely** if v is does not occur in the scope of $\forall v$. If v does not occur freely in φ then we say v is **bound** in φ. If no variable occurs freely in φ then φ is an **$\mathcal{L}$-sentence**.

Remark 2.6. By renaming bound variables, we may assume that no variable v has both free and bound occurrences in the same formula. For example, if φ is the $\mathcal{L}_{or}$-formula $x < y$ and ψ is the $\mathcal{L}_{or}$-formula $\forall x(x + y > x)$, we will write the conjunction $\varphi \wedge \psi$ as $(x < y) \wedge \forall z(z + y > z)$.

We will write $\varphi(v_1, \ldots, v_n)$ to emphasize that φ is an $\mathcal{L}$-formula with free variables $v_1, \ldots, v_n$.

3 Satisfaction

We now define the interpretation of $\mathcal{L}$-formulas as relations on $\mathcal{L}$-structures.

Definition 3.1. Let $\varphi(v_1, \ldots, v_n)$ be an $\mathcal{L}$-formula.

1. Given $\bar{a} \in M^n$, we inductively define what it means for $\bar{a}$ to **satisfy** $\varphi(\bar{v})$ **in** $\mathcal{M}$, written $\mathcal{M} \models \varphi(\bar{a})$.

   (*i*) If $\varphi(v_1, \ldots, v_n)$ is of the form $t_1 = t_2$ where t_1 and t_2 are $\mathcal{L}$-terms using variables among $v_1, \ldots, v_n$, then
   $$\mathcal{M} \models \varphi(\bar{a}) \;\Leftrightarrow\; t_1^{\mathcal{M}}(\bar{a}) = t_2^{\mathcal{M}}(\bar{a}).$$

   (*ii*) If $\varphi(v_1, \ldots, v_n)$ is of the form $R(t_1, \ldots, t_m)$ where R is an m-ary relation symbol and $t_1, \ldots, t_m$ are $\mathcal{L}$-terms with variables among $v_1, \ldots, v_n$ then
   $$\mathcal{M} \models \varphi(\bar{a}) \;\Leftrightarrow\; (t_1^{\mathcal{M}}(\bar{a}), \ldots, t_m^{\mathcal{M}}(\bar{a})) \in R^{\mathcal{M}}.$$

   (*iii*) If $\varphi(v_1, \ldots, v_n)$ is an $\mathcal{L}$-formula then
   $$\mathcal{M} \models \neg\varphi(\bar{a}) \;\Leftrightarrow\; \mathcal{M} \not\models \varphi(\bar{a}).$$

   (*iv*) If $\varphi(v_{i_1}, \ldots, v_{i_r})$ and $\psi(v_{j_1}, \ldots, v_{j_s})$ are $\mathcal{L}$-formulas, with $\{i_1, \ldots, i_r, j_1, \ldots, j_s\} = \{1, \ldots, n\}$, then
   $$\mathcal{M} \models (\varphi \wedge \psi)(\bar{a}) \;\Leftrightarrow\; \mathcal{M} \models \varphi(a_{i_1}, \ldots, a_{i_r}) \text{ and } \mathcal{M} \models \psi(a_{j_1}, \ldots, a_{j_s}),$$

   (*v*) If $\varphi(v_1, \ldots, v_n, w)$ is an $\mathcal{L}$-formula then
   $$\mathcal{M} \models (\forall w \varphi)(\bar{a}) \;\Leftrightarrow\; \mathcal{M} \models \varphi(\bar{a}, b) \text{ for every } b \in M.$$

Remark 3.2.

1. We will use the following abbreviations for the expression of other "logical notions".

   (*i*) *disjunction*: $\varphi \vee \psi$ ("φ or ψ") is an abbreviation for $\neg(\neg\varphi \wedge \neg\psi)$.

   (*ii*) *implication*: $\varphi \rightarrow \psi$ ("φ implies ψ") is an abbreviation for $\neg\varphi \vee \psi$.

   (*iii*) *equivalence*: $\varphi \leftrightarrow \psi$ ("φ if and only if ψ") is an abbreviation for $(\varphi \rightarrow \psi) \wedge (\psi \rightarrow \varphi)$.

   (*iv*) *existential quantification*: $\exists v(\varphi)$ ("there exists v, φ") is an abbreviation for $\neg\forall v(\neg\varphi)$.

2. Depending on the particular language $\mathcal{L}$, one can define further abbreviations. For example, consider the language $\mathcal{L}_{or}$. We often drop the multiplication symbol, and write $v_1 v_2$ for $v_1 \cdot v_2$. We can express the squaring function as the $\mathcal{L}_{or}$-term $v \cdot v$, which will be abbreviated as v^2. For example, the following formula expresses that every positive element has a square root:
   $$\forall x(x > 0 \rightarrow \exists y(x = y^2)).$$

   For another example, we can express the *ternary relation* $|x - y| < z$ as
   $$(0 \leq x - y < z) \vee (0 \leq y - x < z),$$

   where $v_1 \leq v_2 < v_3$ is an abbreviation for: $((v_1 = v_2) \vee (v_1 < v_2)) \wedge (v_2 < v_3)$.

Note that if we have an $\mathcal{L}$-formula $\varphi(v_1, \ldots, v_n)$ and an $\mathcal{L}$-structure $\mathcal{M}$, it only makes sense to ask whether φ is satisfied in $\mathcal{M}$ *after* plugging in elements from the universe of M for the free variables. On the other hand, if a formula has no free variables, then we can think of it as expressing some property that is either true or false in a structure.

Definition 3.3. An **$\mathcal{L}$-sentence** is an $\mathcal{L}$-formula with no free variables.

Definition 3.1 does not technically apply directly to sentences since we started with a formula with free variables. However, you should have an intuitive idea for what it should mean for an $\mathcal{L}$-structure to *satisfy* an $\mathcal{L}$-sentence. Moreover, you should be able to write out a formal definition. On the other hand, we can also "cheat", and say that an $\mathcal{L}$-structure $\mathcal{M}$ **satisfies** an $\mathcal{L}$-sentence φ if, for some/any $a \in M$, a satisfies the formula $\varphi \wedge (v_1 = v_1)$ in $\mathcal{M}$ as in Definition 3.1.

Definition 3.4. If $\mathcal{M}$ is an $\mathcal{L}$-structure and φ is an $\mathcal{L}$-sentence, then we write $\mathcal{M} \models \varphi$ to mean that $\mathcal{M}$ satisfies φ.

In the previous situation, we may also say "$\mathcal{M}$ models φ", or "φ is true in $\mathcal{M}$".

Definition 3.5. Let Σ be a set of $\mathcal{L}$-sentences.

1. Σ is **satisfiable** if there is an $\mathcal{L}$-structure $\mathcal{M}$ such that $\mathcal{M} \models \varphi$ for all $\varphi \in \Sigma$ (we also write $\mathcal{M} \models \Sigma$ in this case).

2. Σ is **finitely satisfiable** if every finite subset of Σ is satisfiable.

Note that if a set of $\mathcal{L}$-sentences is finitely satisfiable, then its finite subsets may very well be satisfied by different $\mathcal{L}$-structures. So there is no obvious reason why a finitely satisfiable set of sentences should be satisfiable. However, this is in fact the case, thanks to the Compactness Theorem, which is the foundation of first-order model theory.

Theorem 3.6 (Compactness)**.** *Let Σ be a set of $\mathcal{L}$-sentences. Then Σ is satisfiable if and only if it is finitely satisfiable.*