2 Structures and first-order logic

2.1 The language of first-order logic

First-order logic is a richer language than propositional logic. Its lexicon contains not only the symbols $\wedge$, $\vee$, $\neg$, $\rightarrow$, and $\leftrightarrow$ (and parentheses) from propositional logic, but also the symbols $\exists$ and $\forall$ for "there exists" and "for all," along with various symbols to represent variables, constants, functions, and relations. These symbols are grouped into five categories.

- **Variables.** Lower case letters from the end of the alphabet $(\ldots x, y, z)$ are used to denote variables. Variables represent arbitrary elements of an underlying set. This, in fact, is what "first-order" refers to. Variables that represent sets of elements are called *second-order*. Second-order logic, discussed in Chapter 9, is distinguished by the inclusion of such variables.
- **Constants.** Lower case letters from the beginning of the alphabet $(a, b, c, \ldots)$ are usually used to denote constants. A constant represents a specific element of an underlying set.
- **Functions.** The lower case letters f, g, and h are commonly used to denote functions. The arguments may be parenthetically listed following the function symbol as $f(x_1, x_2, \ldots, x_n)$. First-order logic has symbols for functions of any number of variables. If f is a function of one, two, or three variables, then it is called *unary*, *binary*, or *ternary*, respectively. In general, a function of n variables is called *n-ary* and n is referred to as the *arity* of the function.
- **Relations.** Capital letters, especially P, Q, R, and S, are used to denote relations. As with functions, each relation has an associated arity.

We have an infinite number of each of these four types of symbols at our disposal. Since there are only finitely many letters, subscripts are used to accomplish this infinitude. For example, x_1, x_2, $x_3, \ldots$ are often used to denote variables. Of course, we can use any symbol we want in first-order logic. Ascribing the letters of the alphabet in the above manner is a convenient convention. If you turn to a random page in this book and see "$R(a, x, y)$," you can safely assume that R is a ternary relation, x and y are variables, and a is a constant. However, we may at times use symbols that we have not yet mentioned. We may use the symbol $\heartsuit$ if we please. However, if we do so, we must say what this symbol represents, whether it is a constant, a variable, a function of 23 variables, a ternary relation, or what.

- **Fixed symbols.** The fixed symbols are $\wedge$, $\vee$, $\neg$, $\rightarrow$, $\leftrightarrow$, (,), $\exists$, and $\forall$.

By "fixed" we mean that these symbols are always interpreted in the same way. If you look on page 211 of this book and see the symbol $\wedge$, it means the same thing as it did on page 8. It means "and." The same is true for each of the fixed symbols. In contrast, the interpretation of the function symbol f depends on the context. We may use this symbol to represent any function we choose.

The fixed symbols $\exists$ and $\forall$, called *quantifiers*, make the language of first-order logic far more expressive than propositional logic. They are called the *existential* and *universal* quantifiers, respectively. In any first-order formula, each quantifier is immediately followed by a variable. We read $\exists x$ as "there exists x such that" and $\forall x$ as "for all x."

The following is an example of a sentence of first-order logic: $\forall y \exists x R(f(x), y)$. This sentence says that for all y there exists x such that the relation R holds for the ordered pair $(f(x), y)$. Here f is a unary function and R is a binary relation. Whether this sentence is true or not depends on the context. If the relation R is equality, then this sentence is true if and only if the function f is onto.

Because of the ubiquity of equality in mathematics, we add to our list of fixed symbols the symbol $=$ for "equals." We still refer to this as "first-order logic" although it is often called "first-order logic with equality." The inclusion of equality allows the quantifiers to actually quantify. For example, the sentence

$$\exists x_1 \exists x_2 \exists x_3 (\neg(x_1 = x_2) \wedge \neg(x_1 = x_3) \wedge \neg(x_2 = x_3))$$

says that there exist at least three distinct elements. Likewise, we can write sentences that say there exist at least seven elements, or fewer than 23 elements, or exactly 45 elements.

We have now completely listed the symbols of first-order logic. Our next objective is to define the syntax and semantics. That is, we need to say which strings of these symbols are permissable as formulas and also how to interpret the formulas.

2.2 The syntax of first-order logic

The definition of a *formula* in first-order logic is analogous to the definition of formula in propositional logic. We first define *atomic formulas* and then give rules for constructing more complex formulas. We used upper case Roman letters such as F, G, and H to denote formulas in propositional logic. In first-order logic, we reserve these letters for other uses and instead use lower case Greek letters such as φ, ψ, and θ to denote formulas.

Prior to defining formulas, we must define the term *term*. Terms are defined inductively by the following two rules.

(T1) Every variable and constant is a term.
(T2) If f is an m-ary function and $t_1, \dots, t_m$ are terms,
then $f(t_1, \dots, t_m)$ is also a term.

Definition 2.1 An *atomic formula* is a formula that has the form $t_1 = t_2$ or $R(t_1, \dots, t_n)$ where R is an n-ary relation and $t_1, \dots, t_n$ are terms.

As with propositional logic, we regard some of the fixed symbols as *primitive*. The other symbols are defined in terms of the primitive symbols. We view $\neg$, $\wedge$, and $\exists$ as primitive. Every formula of first-order logic is built from atomic formulas by repeated application of three rules. Each rule corresponds to a primitive symbol.

(R1) If φ is a formula then so is $\neg\varphi$.
(R2) If φ and ψ are formulas then so is $\varphi \wedge \psi$.
(R3) If φ is a formula, then so is $\exists x \varphi$ for any variable x.

Note that (R1) and (R2) were also rules for propositional logic and only the rule (R3) is new.

Definition 2.2 A string of symbols is a *formula* of first-order logic if and only if it is constructed from atomic formulas by repeated application of rules (R1), (R2), and (R3).

The definitions of $\vee$, $\rightarrow$, and $\leftrightarrow$ are the same as in propositional logic. We define $\forall x \varphi$ as $\neg\exists x \neg\varphi$. For any formula φ, the two formulas $\forall x \varphi$ and $\neg\exists x \neg\varphi$ are interchangeable. So from (R1) and (R3) we have the following: if φ is a formula, then so is $\forall x \varphi$ for any variable x.

Example 2.3 $\forall y P(x,y) \vee \exists y Q(x,y)$ is a formula of first-order logic and $x(Q\forall P)y\exists)(\vee$ is not.

In the next section, we discuss the semantics of first-order logic. For this we need to know the order of operations of the symbols. Parentheses dictate the order of operations in any formula. In absence of parentheses, we use the following rule: $\neg, \exists,$ and $\forall$ have priority over $\wedge, \vee, \rightarrow,$ and $\leftrightarrow$.

Example 2.4 $\exists x P(x,y) \vee Q(x,y)$ means $(\exists x P(x,y)) \vee (Q(x,y))$ and $\forall y P(x,y) \rightarrow Q(x,y)$ means $(\forall y P(x,y)) \rightarrow (Q(x,y))$.

We also use the following convention: the order in which to consider $\neg$, $\exists$, and $\forall$ is determined by the order in which they are listed. We again employ conventions (C1) and (C2) from Section 1.1. These allow us to drop parentheses that are not needed.

Example 2.5 We write $\neg\exists x \forall y \exists z R(x,y,z)$ instead of $\neg(\exists x(\forall y(\exists z(R(x,y,z)))))$.

Having defined formulas, we next define the notion of a subformula.

Definition 2.6 Let φ be a formula of first-order logic. We inductively define what it means for θ to be a *subformula* of φ as follows:

If φ is atomic, then θ is a subformula of φ if and only if $\theta = \varphi$.
If φ has the form $\neg\psi$, then θ is a subformula of φ if and only if $\theta = \varphi$ or θ is a subformula of ψ.
If φ has the form $\psi_1 \wedge \psi_2$, then θ is a subformula of φ if and only if $\theta = \varphi$ or θ is a subformula of ψ_1, or θ is a subformula of ψ_2.
If φ has the form $\exists x \psi$, then θ is a subformula of φ if and only if $\theta = \varphi$ or θ is a subformula of ψ.

Example 2.7 Let φ be the formula $\exists x \forall y P(x,y) \vee \forall x \exists y Q(x,y)$ where P and Q are binary relations. The subformulas of φ are $\exists x \forall y P(x,y)$, $\forall y P(x,y), P(x,y), \forall x \exists y Q(x,y), \exists y Q(x,y), Q(x,y)$ *and* φ *itself*.
Note that the formula $P(x,y) \vee \forall x \exists y Q(x,y)$, occurring as part of φ, is not a subformula of φ.

The *free variables* of a formula φ are those variables occurring in φ that are not quantified. For example, in the formula $\forall y R(x,y)$, x is a free variable, but y is not since it is quantified by $\forall$. For any first-order formula φ, let $free(\varphi)$ denote the set of free variables of φ. We can define $free(\varphi)$ inductively as follows:

If φ is atomic, then $free(\varphi)$ is the set of all variables occurring in φ,
if $\varphi = \neg\psi$, then $free(\varphi) = free(\psi)$,
if $\varphi = \psi \wedge \theta$, then $free(\varphi) = free(\psi) \cup free(\theta)$, and
if $\varphi = \exists x \psi$, then $free(\varphi) = free(\psi) - \{x\}$.

Definition 2.8 A *sentence* of first-order logic is a formula having no free variables.

Example 2.9 $\exists x \forall y P(x,y) \vee \forall x \exists y Q(x,y)$ is a sentence of first-order logic, whereas $\exists x P(x,y) \vee \forall x Q(x,y)$ is a formula but not a sentence (y is a free variable).

Example 2.10 Let φ be the formula $\forall y \exists x f(x) = y$. Then φ is a sentence since both of the variables occurring in φ are quantified. The formulas $f(x) = y$ and $\exists x f(x) = y$ are both subformulas of φ. Neither of these subformulas is a sentence.

In contrast to the free variables of a formula φ, the *bound* variables of φ are those variables that do have quantifiers. For any first-order formula φ, $bnd(\varphi)$ denotes the set of bound variables occurring in φ. Again, this notion can be precisely defined by induction.

If φ is atomic, then $bnd(\varphi) = \emptyset$,
if $\varphi = \neg\psi$, then $bnd(\varphi) = bnd(\psi)$,
if $\varphi = \psi \wedge \theta$, then $bnd(\varphi) = bnd(\psi) \cup bnd(\theta)$, and
if $\varphi = \exists x \psi$, then $bnd(\varphi) = bnd(\psi) \cup \{x\}$.

Every variable occurring in φ is in $free(\varphi)$ or $bnd(\varphi)$. As the next example shows, these two sets are not necessarily disjoint. A variable can have both free and bound occurrences within the same formula.

Example 2.11 Consider the formula $\exists x(R(x,y) \wedge \exists y R(y,x))$. The variable y occurs free in $R(x,y)$ and bound in $\exists y R(y,x)$. The variable x occurs only as a bound variable. So, if ψ denotes this formula, then *free*$(\psi) = \{y\}$ and $bnd(\psi) = \{x,y\}$.

Notation *We write $\varphi(x_1, x_2, \ldots, x_n)$ to denote a formula having free variables $x_1, x_2, \ldots, x_n$. We write $\varphi(t_1, t_2, \ldots, t_n)$ to denote the formula obtained by replacing each free occurrence of x_i in φ with the term t_i. When using this notation, it should always be assumed that each t_i contains none of the variables in bnd(φ). (E.g., if $\varphi(x)$ is $\exists\ _y\neg(x = y)$ then we do not allow the substitution $\varphi(y)$).*

The presence of free variables distinguishes formulas from sentences. This distinction did not exist in propositional logic. The notion of truth is defined only for sentences. It does not make sense to ask whether the formula $y = x + 1$ is true or not. But we can ask whether $\forall y \exists x(y = x + 1)$ or $c_1 = c_2 + 1$ is true or not. The answer, as we have already indicated, depends on the context.

2.3 Semantics and structures

As with propositional logic, the semantics for $\wedge$ and $\neg$ can be described by saying $\wedge$ behaves like "and" and $\neg$ behaves like "negation." Likewise, the semantics for the quantifiers $\exists$ and $\forall$ can be inferred from the phrases "there exists" and "for all." However, we must be more precise when defining the semantics of a logic.