First-order logic

From Wikipedia, the free encyclopedia

First-order logic is a collection of formal systems used in mathematics, philosophy, linguistics, and computer science. It is also known as **first-order predicate calculus**, the **lower predicate calculus**, **quantification theory**, and **predicate logic**. First-order logic uses quantified variables over (non-logical) objects. It allows the use of sentences that contain variables, so that rather than propositions such as *Socrates is a man* one can have expressions in the form *X is a man* where X is a variable.[1] This distinguishes it from propositional logic, which does not use quantifiers.

A theory about a topic is usually a first-order logic together with a specified domain of discourse over which the quantified variables range, finitely many functions from that domain to itself, finitely many predicates defined on that domain, and a set of axioms believed to hold for those things. Sometimes "theory" is understood in a more formal sense, which is just a set of sentences in first-order logic.

The adjective "first-order" distinguishes first-order logic from higher-order logic in which there are predicates having predicates or functions as arguments, or in which one or both of predicate quantifiers or function quantifiers are permitted.[2] In first-order theories, predicates are often associated with sets. In interpreted higher-order theories, predicates may be interpreted as sets of sets.

There are many deductive systems for first-order logic which are both sound (all provable statements are true in all models) and complete (all statements which are true in all models are provable). Although the logical consequence relation is only semidecidable, much progress has been made in automated theorem proving in first-order logic. First-order logic also satisfies several metalogical theorems that make it amenable to analysis in proof theory, such as the Löwenheim–Skolem theorem and the compactness theorem.

First-order logic is the standard for the formalization of mathematics into axioms and is studied in the foundations of mathematics. Peano arithmetic and Zermelo–Fraenkel set theory are axiomatizations of number theory and set theory, respectively, into first-order logic. No first-order theory, however, has the strength to uniquely describe a structure with an infinite domain, such as the natural numbers or the real line. Axioms systems that do fully describe these two structures (that is, categorical axiom systems) can be obtained in stronger logics such as second-order logic.

For a history of first-order logic and how it came to dominate formal logic, see José Ferreirós (2001).

Contents

Introduction

While propositional logic deals with simple declarative propositions, first-order logic additionally covers predicates and quantification.

A predicate takes an entity or entities in the domain of discourse as input and outputs either True or False. Consider the two sentences "Socrates is a philosopher" and "Plato is a philosopher". In propositional logic, these sentences are viewed as being unrelated and might be denoted, for example, by variables such as p and q. The predicate "is a philosopher" occurs in both sentences, which have a common structure of "a is a philosopher". The variable a is instantiated as "Socrates" in the first sentence and is instantiated as "Plato" in the second sentence. While first-order logic allows for the use of predicates, such as "is a philosopher" in this example, propositional logic does not.

Relationships between predicates can be stated using logical connectives. Consider, for example, the first-order formula "if a is a philosopher, then a is a scholar". This formula is a conditional statement with "a is a philosopher" as its hypothesis and "a is a scholar" as its conclusion. The truth of this formula depends on which object is denoted by a, and on the interpretations of the predicates "is a philosopher" and "is a scholar".

Quantifiers can be applied to variables in a formula. The variable a in the previous formula can be universally quantified, for instance, with the first-order sentence "For every a, if a is a philosopher, then a is a scholar". The universal quantifier "for every" in this sentence expresses the idea that the claim "if a is a philosopher, then a is a scholar" holds for *all* choices of a.

The *negation* of the sentence "For every a, if a is a philosopher, then a is a scholar" is logically equivalent to the sentence "There exists a such that a is a philosopher and a is not a scholar". The existential quantifier "there exists" expresses the idea that the claim "a is a philosopher and a is not a scholar" holds for *some* choice of a.

The predicates "is a philosopher" and "is a scholar" each take a single variable. In general, predicates can take several variables. In the first-order sentence "Socrates is the teacher of Plato", the predicate "is the teacher of" takes two variables.

An interpretation (or model) of a first-order formula specifies what each predicate means and the entities that can instantiate the variables. These entities form the domain of discourse or universe, which is usually required to be a nonempty set. For example, in interpretation with the domain of discourse consisting of all human beings and the predicate "is a philosopher" understood as "was the author of the *Republic*", the sentence "There exists a such that a is a philosopher" is seen as being true, as witnessed by Plato.

Syntax

There are two key parts of first-order logic. The syntax determines which collections of symbols are legal expressions in first-order logic, while the semantics determine the meanings behind these expressions.

Alphabet

Unlike natural languages, such as English, the language of first-order logic is completely formal, so that it can be mechanically determined whether a given expression is legal. There are two key types of legal expressions: **terms**, which intuitively represent objects, and **formulas**, which intuitively express predicates that can be true or false. The terms and formulas of first-order logic are strings of **symbols** which together form the **alphabet** of the language. As with all formal languages, the nature of the symbols themselves is outside the scope of formal logic; they are often regarded simply as letters and punctuation symbols.

It is common to divide the symbols of the alphabet into **logical symbols**, which always have the same meaning, and **non-logical symbols**, whose meaning varies by interpretation. For example, the logical symbol $\land$ always represents "and"; it is never interpreted as "or". On the other hand, a non-logical predicate symbol such as Phil(x) could be interpreted to mean "x is a philosopher", "x is a man named Philip", or any other unary predicate, depending on the interpretation at hand.

Logical symbols

There are several logical symbols in the alphabet, which vary by author but usually include:

- The quantifier symbols ∀ and ∃
- The logical connectives: ∧ for conjunction, ∨ for disjunction, → for implication, ↔ for biconditional, ¬ for negation. Occasionally other logical connective symbols are included. Some authors use Cpq, instead of →, and Epq, instead of ↔, especially in contexts where → is used for other purposes. Moreover, the horseshoe ⊃ may replace →; the triple-bar ≡ may replace ↔; a tilde (~), Np, or Fpq, may replace ¬; ||, or Apq may replace ∨; and &, Kpq, or the middle dot, ·, may replace ∧, especially if these symbols are not available for technical reasons. (*Note*: the aforementioned symbols Cpq, Epq, Np, Apq, and Kpq are used in Polish notation.)
- Parentheses, brackets, and other punctuation symbols. The choice of such symbols varies depending on context.
- An infinite set of **variables**, often denoted by lowercase letters at the end of the alphabet x, y, z, ... Subscripts are often used to distinguish variables: x_0, x_1, x_2, ...
- An **equality symbol** (sometimes, **identity symbol**) =; see the section on equality below.

It should be noted that not all of these symbols are required – only one of the quantifiers, negation and conjunction, variables, brackets and equality suffice. There are numerous minor variations that may define additional logical symbols:

- Sometimes the truth constants T, Vpq, or ⊤, for "true" and F, Opq, or ⊥, for "false" are included. Without any such logical operators of valence 0, these two constants can only be expressed using quantifiers.
- Sometimes additional logical connectives are included, such as the Sheffer stroke, Dpq (NAND), and exclusive or, Jpq.

Non-logical symbols

The non-logical symbols represent predicates (relations), functions and constants on the domain of discourse. It used to be standard practice to use a fixed, infinite set of non-logical symbols for all purposes. A more recent practice is to use different non-logical symbols according to the application one has in mind. Therefore, it has become necessary to name the set of all non-logical symbols used in a particular application. This choice is made via a **signature**.[3]

The traditional approach is to have only one, infinite, set of non-logical symbols (one signature) for all applications. Consequently, under the traditional approach there is only one language of first-order logic.[4] This approach is still common, especially in philosophically oriented books.

1. For every integer $n \geq 0$ there is a collection of n-**ary**, or n-**place**, **predicate symbols**. Because they represent relations between n elements, they are also called **relation symbols**. For each arity n we have an infinite supply of them:
   $P^n_0, P^n_1, P^n_2, P^n_3, ...$
2. For every integer $n \geq 0$ there are infinitely many n-ary **function symbols**:
   $f^n_0, f^n_1, f^n_2, f^n_3, ...$

In contemporary mathematical logic, the signature varies by application. Typical signatures in mathematics are {1, ×} or just {×} for groups, or {0, 1, +, ×, <} for ordered fields. There are no restrictions on the number of non-logical symbols. The signature can be empty, finite, or infinite, even uncountable. Uncountable signatures occur for example in modern proofs of the Löwenheim-Skolem theorem.

In this approach, every non-logical symbol is of one of the following types.

1. A **predicate symbol** (or **relation symbol**) with some **valence** (or **arity**, number of arguments) greater than or equal to 0. These are often denoted by uppercase letters P, Q, R,... .
   - Relations of valence 0 can be identified with propositional variables. For example, P, which can stand for any statement.
   - For example, $P(x)$ is a predicate variable of valence 1. One possible interpretation is "x is a man".
   - $Q(x,y)$ is a predicate variable of valence 2. Possible interpretations include "x is greater than y" and "x is the father of y".
2. A **function symbol**, with some valence greater than or equal to 0. These are often denoted by lowercase letters f, g, h,... .
   - Examples: $f(x)$ may be interpreted as for "the father of x". In arithmetic, it may stand for "-x". In set theory, it may stand for "the power set of x". In arithmetic, $g(x,y)$ may stand for "$x+y$". In set theory, it may stand for "the union of x and y".
   - Function symbols of valence 0 are called **constant symbols**, and are often denoted by lowercase letters at the beginning of the alphabet a, b, c,... . The symbol a may stand for Socrates. In arithmetic, it may stand for 0. In set theory, such a constant may stand for the empty set.

The traditional approach can be recovered in the modern approach by simply specifying the "custom" signature to consist of the traditional sequences of non-logical symbols.

Formation rules

The formation rules define the terms and formulas of first order logic. When terms and formulas are represented as strings of symbols, these rules can be used to write a formal grammar for terms and formulas. These rules are generally context-free (each production has a single symbol on the left side), except that the set of symbols may be allowed to be infinite and there may be many start symbols, for example the variables in the case of terms.

Terms

The set of **terms** is inductively defined by the following rules:

1. **Variables.** Any variable is a term.
2. **Functions.** Any expression $f(t_1,...,t_n)$ of n arguments (where each argument t_i is a term and f is a function symbol of valence n) is a term.

In particular, symbols denoting individual constants are 0-ary function symbols, and are thus terms.

Only expressions which can be obtained by finitely many applications of rules 1 and 2 are terms. For example, no expression involving a predicate symbol is a term.

Formulas

The set of **formulas** (also called well-formed formulas [5] or **wff**s) is inductively defined by the following rules:

1. **Predicate symbols.** If P is an n-ary predicate symbol and t_1, ..., t_n are terms then $P(t_1,...,t_n)$ is a formula.
2. **Equality.** If the equality symbol is considered part of logic, and t_1 and t_2 are terms, then $t_1 = t_2$ is a formula.
3. **Negation.** If φ is a formula, then $\neg\varphi$ is a formula.
4. **Binary connectives.** If φ and ψ are formulas, then $(\varphi \to \psi)$ is a formula. Similar rules apply to other binary logical connectives.
5. **Quantifiers.** If φ is a formula and x is a variable, then $\forall x \varphi$ (for all x, φ holds) and $\exists x \varphi$ (there exists x such that φ) are formulas.

Only expressions which can be obtained by finitely many applications of rules 1–5 are formulas. The formulas obtained from the first two rules are said to be **atomic formulas**.

For example,

$$\forall x \forall y (P(f(x)) \to \neg(P(x) \to Q(f(y), x, z)))$$

is a formula, if f is a unary function symbol, P a unary predicate symbol, and Q a ternary predicate symbol. On the other hand, $\forall x \, x \to$ is not a formula, although it is a string of symbols from the alphabet.

The role of the parentheses in the definition is to ensure that any formula can only be obtained in one way by following the inductive definition (in other words, there is a unique parse tree for each formula). This property is known as **unique readability** of formulas. There are many conventions for where parentheses are used in formulas. For example, some authors use colons or full stops instead of parentheses, or change the places in which parentheses are inserted. Each author's particular definition must be accompanied by a proof of unique readability.

This definition of a formula does not support defining an if-then-else function `ite(c, a, b)`, where "c" is a condition expressed as a formula, that would return "a" if c is true, and "b" if it is false. This is because both predicates and functions can only accept terms as parameters, but the first parameter is a formula. Some languages built on first-order logic, such as SMT-LIB 2.0, add this.[6]

Notational conventions

For convenience, conventions have been developed about the precedence of the logical operators, to avoid the need to write parentheses in some cases. These rules are similar to the order of operations in arithmetic. A common convention is:

- $\neg$ is evaluated first
- $\land$ and $\lor$ are evaluated next
- Quantifiers are evaluated next
- $\to$ is evaluated last.

Moreover, extra punctuation not required by the definition may be inserted to make formulas easier to read. Thus the formula

$$(\neg \forall x P(x) \to \exists x \neg P(x))$$

might be written as

$$(\neg [\forall x P(x)]) \to \exists x [\neg P(x)].$$

In some fields, it is common to use infix notation for binary relations and functions, instead of the prefix notation defined above. For example, in arithmetic, one typically writes "2 + 2 = 4" instead of "=(+(2,2),4)". It is common to regard formulas in infix notation as abbreviations for the corresponding formulas in prefix notation.

The definitions above use infix notation for binary connectives such as $\to$. A less common convention is Polish notation, in which one writes $\to$, $\land$, and so on in front of their arguments rather than between them. This convention allows all punctuation symbols to be discarded. Polish notation is compact and elegant, but rarely used in practice because it is hard for humans to read it. In Polish notation, the formula

$$\forall x \forall y (P(f(x)) \to \neg(P(x) \to Q(f(y), x, z)))$$

becomes "$\forall x \forall y \to Pfx \neg \to PxQfyxz$".

Free and bound variables

In a formula, a variable may occur **free** or **bound**. Intuitively, a variable is free in a formula if it is not quantified: in $\forall y \, P(x, y)$, variable x is free while y is bound. The free and bound variables of a formula are defined inductively as follows.

1. **Atomic formulas.** If φ is an atomic formula then x is free in φ if and only if x occurs in φ. Moreover, there are no bound variables in any atomic formula.
2. **Negation.** x is free in $\neg\varphi$ if and only if x is free in φ. x is bound in $\neg\varphi$ if and only if x is bound in φ.

3. **Binary connectives.** x is free in $(\varphi \rightarrow \psi)$ if and only if x is free in either φ or ψ. x is bound in $(\varphi \rightarrow \psi)$ if and only if x is bound in either φ or ψ. The same rule applies to any other binary connective in place of $\rightarrow$.

4. **Quantifiers.** x is free in $\forall y\, \varphi$ if and only if x is free in φ and x is a different symbol from y. Also, x is bound in $\forall y\, \varphi$ if and only if x is y or x is bound in φ. The same rule holds with $\exists$ in place of $\forall$.

For example, in $\forall x\, \forall y\, (P(x) \rightarrow Q(x,f(x),z))$, x and y are bound variables, z is a free variable, and w is neither because it does not occur in the formula.

Free and bound variables of a formula need not be disjoint sets: x is both free and bound in $P(x) \rightarrow \forall x\, Q(x)$.

Freeness and boundness can be also specialized to specific occurrences of variables in a formula. For example, in $P(x) \rightarrow \forall x\, Q(x)$, the first occurrence of x is free while the second is bound. In other words, the x in $P(x)$ is free while the x in $\forall x\, Q(x)$ is bound.

A formula in first-order logic with no free variables is called a **first-order sentence**. These are the formulas that will have well-defined truth values under an interpretation. For example, whether a formula such as Phil(x) is true must depend on what x represents. But the sentence $\exists x\, \mathbf{Phil}(x)$ will be either true or false in a given interpretation.

Examples

Ordered abelian groups

In mathematics the language of ordered abelian groups has one constant symbol 0, one unary function symbol $-$, one binary function symbol $+$, and one binary relation symbol $\leq$. Then:
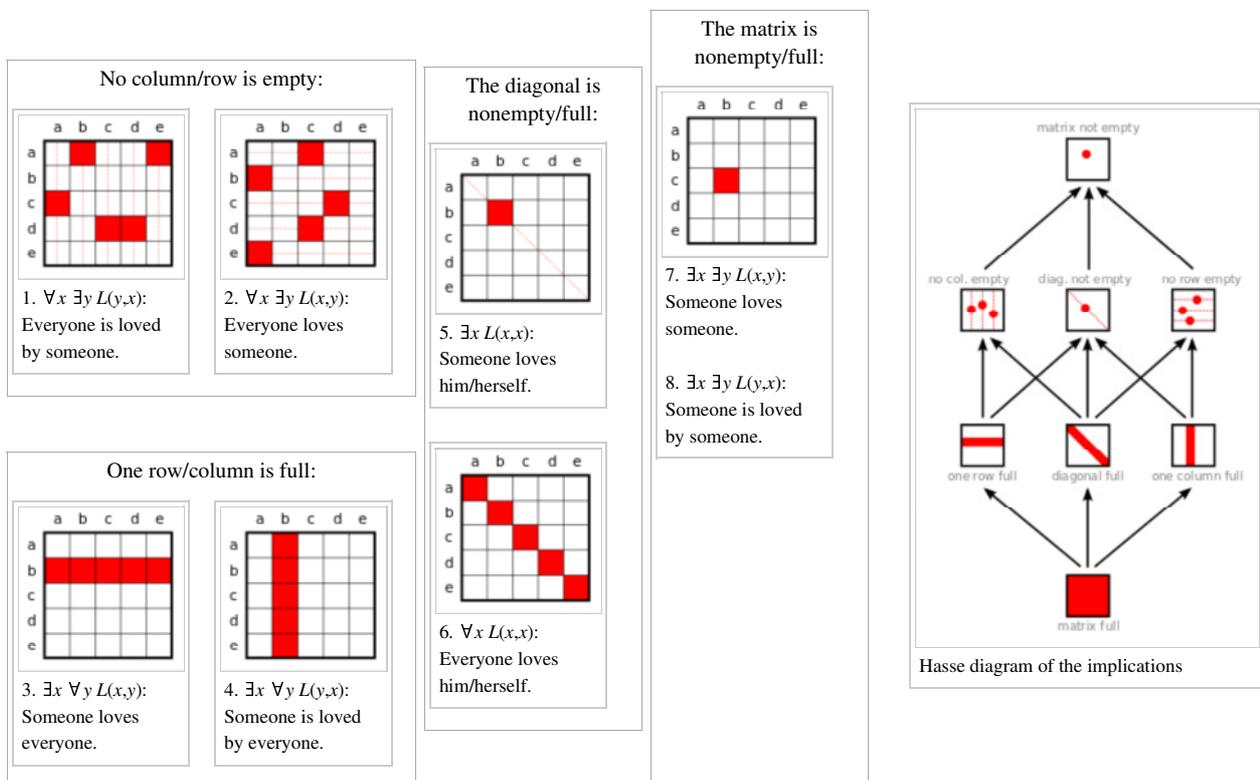
- The expressions $+(x, y)$ and $+(x, +(y, -(z)))$ are **terms**. These are usually written as $x + y$ and $x + y - z$.
- The expressions $+(x, y) = 0$ and $\leq(+(x, +(y, -(z))), +(x, y))$ are **atomic formulas**.

  These are usually written as $x + y = 0$ and $x + y - z \leq x + y$.

- The expression $\left(\forall x \forall y\, \leq(+(x,y), z) \rightarrow \forall x\, \forall y\, +(x,y) = 0\right)$ is a **formula**, which is usually written as $\forall x \forall y (x + y \leq z) \rightarrow \forall x \forall y (x + y = 0)$.

Loving relation

English sentences like "everyone loves someone" can be formalized by first-order logic formulas like $\forall x \exists y\, L(x,y)$. This is accomplished by abbreviating the relation "x loves y" by $L(x,y)$. Using just the two quantifiers $\forall$ and $\exists$ and the loving relation symbol L, but no logical connectives and no function symbols (including constants), formulas with 8 different meanings can be built. The following diagrams show models for each of them, assuming that there are exactly five individuals $a,...,e$ who can love (vertical axis) and be loved (horizontal axis). A small red box at row x and column y indicates $L(x,y)$. Only for the formulas 9 and 10 is the model unique, all other formulas may be satisfied by several models.

No column/row is empty:

1. $\forall x\, \exists y\, L(y,x)$: Everyone is loved by someone.

2. $\forall x\, \exists y\, L(x,y)$: Everyone loves someone.

One row/column is full:

3. $\exists x\, \forall y\, L(x,y)$: Someone loves everyone.

4. $\exists x\, \forall y\, L(y,x)$: Someone is loved by everyone.

The diagonal is nonempty/full:

5. $\exists x\, L(x,x)$: Someone loves him/herself.

6. $\forall x\, L(x,x)$: Everyone loves him/herself.

The matrix is nonempty/full:

7. $\exists x\, \exists y\, L(x,y)$: Someone loves someone.

8. $\exists x\, \exists y\, L(y,x)$: Someone is loved by someone.

Hasse diagram of the implications

9. $\forall x \, \forall y \, L(x,y)$:
Everyone loves
everyone.

10. $\forall x \, \forall y \, L(y,x)$:
Everyone is loved
by everyone.

Each model, represented by a logical matrix, satisfies the formulas in its caption in a "minimal" way, i.e. whitening any red cell in any matrix would make it non-satisfying the corresponding formula. For example, formula 1 is also satisfied by the matrices at 3, 6, and 10, but not by those at 2, 4, 5, and 7. Conversely, the matrix shown at 6 satisfies 1, 2, 5, 6, 7, and 8, but not 3, 4, 9, and 10.

Some formulas imply others, i.e. *all* matrices satisfying the antecedent (LHS) also satisfy the conclusion (RHS) of the implication — e.g. formula 3 implies formula 1, i.e.: each matrix fulfilling formula 3 also fulfills formula 1, but not vice versa (see the Hasse diagram for this ordering relation). In contrast, only some matrices,[7] which satisfy formula 2, happen to satisfy also formula 5, whereas others,[8] also satisfying formula 2, do not; therefore formula 5 is not a logical consequence of formula 2.

The sequence of the quantifiers is important! So it is instructive to distinguish formulas 1: $\forall x \, \exists y \, L(y,x)$, and 3: $\exists x \, \forall y \, L(x,y)$. In both cases everyone is loved; but in the first case everyone (x) is loved by someone (y), in the second case everyone (y) is loved by just exactly one person (x).

Semantics

An interpretation of a first-order language assigns a denotation to all non-logical constants in that language. It also determines a domain of discourse that specifies the range of the quantifiers. The result is that each term is assigned an object that it represents, and each sentence is assigned a truth value. In this way, an interpretation provides semantic meaning to the terms and formulas of the language. The study of the interpretations of formal languages is called formal semantics. What follows is a description of the standard or Tarskian semantics for first-order logic. (It is also possible to define game semantics for first-order logic, but aside from requiring the axiom of choice, game semantics agree with Tarskian semantics for first-order logic, so game semantics will not be elaborated herein.)

The domain of discourse D is a nonempty set of "objects" of some kind. Intuitively, a first-order formula is a statement about these objects; for example, $\exists x P(x)$ states the existence of an object x such that the predicate P is true where referred to it. The domain of discourse is the set of considered objects. For example, one can take D to be the set of integer numbers.

The interpretation of a function symbol is a function. For example, if the domain of discourse consists of integers, a function symbol f of arity 2 can be interpreted as the function that gives the sum of its arguments. In other words, the symbol f is associated with the function $I(f)$ which, in this interpretation, is addition.

The interpretation of a constant symbol is a function from the one-element set D^0 to D, which can be simply identified with an object in D. For example, an interpretation may assign the value $I(c) = 10$ to the constant symbol c.

The interpretation of an *n*-ary predicate symbol is a set of *n*-tuples of elements of the domain of discourse. This means that, given an interpretation, a predicate symbol, and *n* elements of the domain of discourse, one can tell whether the predicate is true of those elements according to the given interpretation. For example, an interpretation $I(P)$ of a binary predicate symbol P may be the set of pairs of integers such that the first one is less than the second. According to this interpretation, the predicate P would be true if its first argument is less than the second.

First-order structures

The most common way of specifying an interpretation (especially in mathematics) is to specify a **structure** (also called a **model**; see below). The structure consists of a nonempty set D that forms the domain of discourse and an interpretation I of the non-logical terms of the signature. This interpretation is itself a function:

- Each function symbol f of arity n is assigned a function $I(f)$ from D^n to D. In particular, each constant symbol of the signature is assigned an individual in the domain of discourse.
- Each predicate symbol P of arity n is assigned a relation $I(P)$ over D^n or, equivalently, a function from D^n to $\{\mathbf{true}, \mathbf{false}\}$. Thus each predicate symbol is interpreted by a Boolean-valued function on D.

Evaluation of truth values

A formula evaluates to true or false given an interpretation, and a **variable assignment** μ that associates an element of the domain of discourse with each variable. The reason that a variable assignment is required is to give meanings to formulas with free variables, such as $y = x$. The

truth value of this formula changes depending on whether x and y denote the same individual.

First, the variable assignment μ can be extended to all terms of the language, with the result that each term maps to a single element of the domain of discourse. The following rules are used to make this assignment:

1. **Variables.** Each variable x evaluates to $\mu(x)$
2. **Functions.** Given terms $t_1, \ldots, t_n$ that have been evaluated to elements $d_1, \ldots, d_n$ of the domain of discourse, and a n-ary function symbol f, the term $f(t_1, \ldots, t_n)$ evaluates to $(I(f))(d_1, \ldots, d_n)$.

Next, each formula is assigned a truth value. The inductive definition used to make this assignment is called the T-schema.

1. **Atomic formulas (1).** A formula $P(t_1, \ldots, t_n)$ is associated the value true or false depending on whether $\langle v_1, \ldots, v_n \rangle \in I(P)$, where $v_1, \ldots, v_n$ are the evaluation of the terms $t_1, \ldots, t_n$ and $I(P)$ is the interpretation of P, which by assumption is a subset of D^n.
2. **Atomic formulas (2).** A formula $t_1 = t_2$ is assigned true if t_1 and t_2 evaluate to the same object of the domain of discourse (see the section on equality below).
3. **Logical connectives.** A formula in the form $\neg\phi$, $\phi \rightarrow \psi$, etc. is evaluated according to the truth table for the connective in question, as in propositional logic.
4. **Existential quantifiers.** A formula $\exists x \phi(x)$ is true according to M and μ if there exists an evaluation μ' of the variables that only differs from μ regarding the evaluation of x and such that φ is true according to the interpretation M and the variable assignment μ'. This formal definition captures the idea that $\exists x \phi(x)$ is true if and only if there is a way to choose a value for x such that φ(x) is satisfied.
5. **Universal quantifiers.** A formula $\forall x \phi(x)$ is true according to M and μ if φ(x) is true for every pair composed by the interpretation M and some variable assignment μ' that differs from μ only on the value of x. This captures the idea that $\forall x \phi(x)$ is true if every possible choice of a value for x causes φ(x) to be true.

If a formula does not contain free variables, and so is a sentence, then the initial variable assignment does not affect its truth value. In other words, a sentence is true according to M and μ if and only if it is true according to M and every other variable assignment μ'.

There is a second common approach to defining truth values that does not rely on variable assignment functions. Instead, given an interpretation M, one first adds to the signature a collection of constant symbols, one for each element of the domain of discourse in M; say that for each d in the domain the constant symbol c_d is fixed. The interpretation is extended so that each new constant symbol is assigned to its corresponding element of the domain. One now defines truth for quantified formulas syntactically, as follows:

1. **Existential quantifiers (alternate).** A formula $\exists x \phi(x)$ is true according to M if there is some d in the domain of discourse such that $\phi(c_d)$ holds. Here $\phi(c_d)$ is the result of substituting c_d for every free occurrence of x in φ.
2. **Universal quantifiers (alternate).** A formula $\forall x \phi(x)$ is true according to M if, for every d in the domain of discourse, $\phi(c_d)$ is true according to M.

This alternate approach gives exactly the same truth values to all sentences as the approach via variable assignments.

Validity, satisfiability, and logical consequence

If a sentence φ evaluates to True under a given interpretation M, one says that M **satisfies** φ; this is denoted $M \models \varphi$. A sentence is **satisfiable** if there is some interpretation under which it is true.

Satisfiability of formulas with free variables is more complicated, because an interpretation on its own does not determine the truth value of such a formula. The most common convention is that a formula with free variables is said to be satisfied by an interpretation if the formula remains true regardless which individuals from the domain of discourse are assigned to its free variables. This has the same effect as saying that a formula is satisfied if and only if its universal closure is satisfied.

A formula is **logically valid** (or simply **valid**) if it is true in every interpretation. These formulas play a role similar to tautologies in propositional logic.

A formula φ is a **logical consequence** of a formula ψ if every interpretation that makes ψ true also makes φ true. In this case one says that φ is logically implied by ψ.

Algebraizations

An alternate approach to the semantics of first-order logic proceeds via abstract algebra. This approach generalizes the Lindenbaum–Tarski algebras of propositional logic. There are three ways of eliminating quantified variables from first-order logic that do not involve replacing quantifiers with other variable binding term operators:

- Cylindric algebra, by Alfred Tarski and his coworkers;
- Polyadic algebra, by Paul Halmos;
- Predicate functor logic, mainly due to Willard Quine.

These algebras are all lattices that properly extend the two-element Boolean algebra.

Tarski and Givant (1987) showed that the fragment of first-order logic that has no atomic sentence lying in the scope of more than three quantifiers has the same expressive power as relation algebra. This fragment is of great interest because it suffices for Peano arithmetic and most axiomatic set theory, including the canonical ZFC. They also prove that first-order logic with a primitive ordered pair is equivalent to a relation algebra with two ordered pair projection functions.

First-order theories, models, and elementary classes

A **first-order theory** of a particular signature is a set of axioms, which are sentences consisting of symbols from that signature. The set of axioms is often finite or recursively enumerable, in which case the theory is called **effective**. Some authors require theories to also include all logical consequences of the axioms. The axioms are considered to hold within the theory and from them other sentences that hold within the theory can be derived.

A first-order structure that satisfies all sentences in a given theory is said to be a **model** of the theory. An **elementary class** is the set of all structures satisfying a particular theory. These classes are a main subject of study in model theory.

Many theories have an **intended interpretation**, a certain model that is kept in mind when studying the theory. For example, the intended interpretation of Peano arithmetic consists of the usual natural numbers with their usual operations. However, the Löwenheim–Skolem theorem shows that most first-order theories will also have other, nonstandard models.

A theory is **consistent** if it is not possible to prove a contradiction from the axioms of the theory. A theory is **complete** if, for every formula in its signature, either that formula or its negation is a logical consequence of the axioms of the theory. Gödel's incompleteness theorem shows that effective first-order theories that include a sufficient portion of the theory of the natural numbers can never be both consistent and complete.

For more information on this subject see List of first-order theories and Theory (mathematical logic)

Empty domains

The definition above requires that the domain of discourse of any interpretation must be a nonempty set. There are settings, such as inclusive logic, where empty domains are permitted. Moreover, if a class of algebraic structures includes an empty structure (for example, there is an empty poset), that class can only be an elementary class in first-order logic if empty domains are permitted or the empty structure is removed from the class.

There are several difficulties with empty domains, however:

- Many common rules of inference are only valid when the domain of discourse is required to be nonempty. One example is the rule stating that $\phi \lor \exists x \psi$ implies $\exists x (\phi \lor \psi)$ when x is not a free variable in φ. This rule, which is used to put formulas into prenex normal form, is sound in nonempty domains, but unsound if the empty domain is permitted.
- The definition of truth in an interpretation that uses a variable assignment function cannot work with empty domains, because there are no variable assignment functions whose range is empty. (Similarly, one cannot assign interpretations to constant symbols.) This truth definition requires that one must select a variable assignment function (μ above) before truth values for even atomic formulas can be defined. Then the truth value of a sentence is defined to be its truth value under any variable assignment, and it is proved that this truth value does not depend on which assignment is chosen. This technique does not work if there are no assignment functions at all; it must be changed to accommodate empty domains.

Thus, when the empty domain is permitted, it must often be treated as a special case. Most authors, however, simply exclude the empty domain by definition.

Deductive systems

A **deductive system** is used to demonstrate, on a purely syntactic basis, that one formula is a logical consequence of another formula. There are many such systems for first-order logic, including Hilbert-style deductive systems, natural deduction, the sequent calculus, the tableaux method, and resolution. These share the common property that a deduction is a finite syntactic object; the format of this object, and the way it is constructed, vary widely. These finite deductions themselves are often called **derivations** in proof theory. They are also often called proofs, but are completely formalized unlike natural-language mathematical proofs.

A deductive system is **sound** if any formula that can be derived in the system is logically valid. Conversely, a deductive system is **complete** if every logically valid formula is derivable. All of the systems discussed in this article are both sound and complete. They also share the property that it is possible to effectively verify that a purportedly valid deduction is actually a deduction; such deduction systems are called **effective**.

A key property of deductive systems is that they are purely syntactic, so that derivations can be verified without considering any interpretation. Thus a sound argument is correct in every possible interpretation of the language, regardless whether that interpretation is about mathematics, economics, or some other area.

In general, logical consequence in first-order logic is only semidecidable: if a sentence A logically implies a sentence B then this can be discovered (for example, by searching for a proof until one is found, using some effective, sound, complete proof system). However, if A does not logically imply B, this does not mean that A logically implies the negation of B. There is no effective procedure that, given formulas A and B, always correctly decides whether A logically implies B.

Rules of inference

A **rule of inference** states that, given a particular formula (or set of formulas) with a certain property as a hypothesis, another specific formula (or set of formulas) can be derived as a conclusion. The rule is sound (or truth-preserving) if it preserves validity in the sense that whenever any interpretation satisfies the hypothesis, that interpretation also satisfies the conclusion.

For example, one common rule of inference is the **rule of substitution**. If t is a term and φ is a formula possibly containing the variable x, then $\varphi[t/x]$ is the result of replacing all free instances of x by t in φ. The substitution rule states that for any φ and any term t, one can conclude $\varphi[t/x]$

from φ provided that no free variable of t becomes bound during the substitution process. (If some free variable of t becomes bound, then to substitute t for x it is first necessary to change the bound variables of φ to differ from the free variables of t.)

To see why the restriction on bound variables is necessary, consider the logically valid formula φ given by $\exists x(x = y)$, in the signature of $(0,1,+,\times,=)$ of arithmetic. If t is the term "x + 1", the formula φ[t/y] is $\exists x(x = x + 1)$, which will be false in many interpretations. The problem is that the free variable x of t became bound during the substitution. The intended replacement can be obtained by renaming the bound variable x of φ to something else, say z, so that the formula after substitution is $\exists z(z = x + 1)$, which is again logically valid.

The substitution rule demonstrates several common aspects of rules of inference. It is entirely syntactical; one can tell whether it was correctly applied without appeal to any interpretation. It has (syntactically defined) limitations on when it can be applied, which must be respected to preserve the correctness of derivations. Moreover, as is often the case, these limitations are necessary because of interactions between free and bound variables that occur during syntactic manipulations of the formulas involved in the inference rule.

Hilbert-style systems and natural deduction

A deduction in a Hilbert-style deductive system is a list of formulas, each of which is a **logical axiom**, a hypothesis that has been assumed for the derivation at hand, or follows from previous formulas via a rule of inference. The logical axioms consist of several axiom schemas of logically valid formulas; these encompass a significant amount of propositional logic. The rules of inference enable the manipulation of quantifiers. Typical Hilbert-style systems have a small number of rules of inference, along with several infinite schemas of logical axioms. It is common to have only modus ponens and universal generalization as rules of inference.

Natural deduction systems resemble Hilbert-style systems in that a deduction is a finite list of formulas. However, natural deduction systems have no logical axioms; they compensate by adding additional rules of inference that can be used to manipulate the logical connectives in formulas in the proof.

Sequent calculus

The sequent calculus was developed to study the properties of natural deduction systems. Instead of working with one formula at a time, it uses **sequents**, which are expressions of the form

$$A_1, \ldots, A_n \vdash B_1, \ldots, B_k,$$

where A_1, ..., A_n, B_1, ..., B_k are formulas and the turnstile symbol $\vdash$ is used as punctuation to separate the two halves. Intuitively, a sequent expresses the idea that $(A_1 \wedge \cdots \wedge A_n)$ implies $(B_1 \vee \cdots \vee B_k)$.

Tableaux method

Unlike the methods just described, the derivations in the tableaux method are not lists of formulas. Instead, a derivation is a tree of formulas. To show that a formula A is provable, the tableaux method attempts to demonstrate that the negation of A is unsatisfiable. The tree of the derivation has $\neg A$ at its root; the tree branches in a way that reflects the structure of the formula. For example, to show that $C \vee D$ is unsatisfiable requires showing that C and D are each unsatisfiable; this corresponds to a branching point in the tree with parent $C \vee D$ and children C and D.

A tableaux proof for the propositional formula $((a \vee \neg b) \wedge b) \rightarrow a$.

Resolution

The resolution rule is a single rule of inference that, together with unification, is sound and complete for first-order logic. As with the tableaux method, a formula is proved by showing that the negation of the formula is unsatisfiable. Resolution is commonly used in automated theorem proving.

The resolution method works only with formulas that are disjunctions of atomic formulas; arbitrary formulas must first be converted to this form through Skolemization. The resolution rule states that from the hypotheses $A_1 \vee \cdots \vee A_k \vee C$ and $B_1 \vee \cdots \vee B_l \vee \neg C$, the conclusion $A_1 \vee \cdots \vee A_k \vee B_1 \vee \cdots \vee B_l$ can be obtained.

Provable identities

The following sentences can be called "identities" because the main connective in each is the biconditional.

$$\neg \forall x\, P(x) \Leftrightarrow \exists x\, \neg P(x)$$
$$\neg \exists x\, P(x) \Leftrightarrow \forall x\, \neg P(x)$$
$$\forall x\, \forall y\, P(x, y) \Leftrightarrow \forall y\, \forall x\, P(x, y)$$
$$\exists x\, \exists y\, P(x, y) \Leftrightarrow \exists y\, \exists x\, P(x, y)$$
$$\forall x\, P(x) \wedge \forall x\, Q(x) \Leftrightarrow \forall x\, (P(x) \wedge Q(x))$$
$$\exists x\, P(x) \vee \exists x\, Q(x) \Leftrightarrow \exists x\, (P(x) \vee Q(x))$$
$$P \wedge \exists x\, Q(x) \Leftrightarrow \exists x\, (P \wedge Q(x)) \quad \text{(where } x \text{ must not occur free in } P)$$
$$P \vee \forall x\, Q(x) \Leftrightarrow \forall x\, (P \vee Q(x)) \quad \text{(where } x \text{ must not occur free in } P)$$

Equality and its axioms

There are several different conventions for using equality (or identity) in first-order logic. The most common convention, known as **first-order**
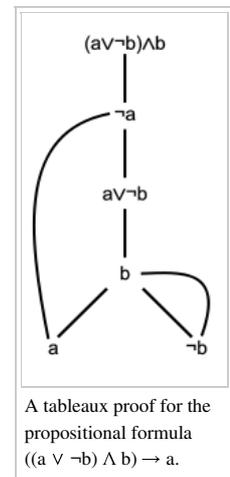
logic with equality, includes the equality symbol as a primitive logical symbol which is always interpreted as the real equality relation between members of the domain of discourse, such that the "two" given members are the same member. This approach also adds certain axioms about equality to the deductive system employed. These equality axioms are:

1. **Reflexivity**. For each variable x, $x = x$.
2. **Substitution for functions.** For all variables x and y, and any function symbol f,
   $$x = y \rightarrow f(...,x,...) = f(...,y,...).$$
3. **Substitution for formulas**. For any variables x and y and any formula $\varphi(x)$, if φ' is obtained by replacing any number of free occurrences of x in φ with y, such that these remain free occurrences of y, then
   $$x = y \rightarrow (\varphi \rightarrow \varphi').$$

These are axiom schemas, each of which specifies an infinite set of axioms. The third schema is known as **Leibniz's law**, "the principle of substitutivity", "the indiscernibility of identicals", or "the replacement property". The second schema, involving the function symbol f, is (equivalent to) a special case of the third schema, using the formula

$$x = y \rightarrow (f(...,x,...) = z \rightarrow f(...,y,...) = z).$$

Many other properties of equality are consequences of the axioms above, for example:

1. **Symmetry.** If $x = y$ then $y = x$.
2. **Transitivity.** If $x = y$ and $y = z$ then $x = z$.

First-order logic without equality

An alternate approach considers the equality relation to be a non-logical symbol. This convention is known as **first-order logic without equality**. If an equality relation is included in the signature, the axioms of equality must now be added to the theories under consideration, if desired, instead of being considered rules of logic. The main difference between this method and first-order logic with equality is that an interpretation may now interpret two distinct individuals as "equal" (although, by Leibniz's law, these will satisfy exactly the same formulas under any interpretation). That is, the equality relation may now be interpreted by an arbitrary equivalence relation on the domain of discourse that is congruent with respect to the functions and relations of the interpretation.

When this second convention is followed, the term **normal model** is used to refer to an interpretation where no distinct individuals a and b satisfy $a = b$. In first-order logic with equality, only normal models are considered, and so there is no term for a model other than a normal model. When first-order logic without equality is studied, it is necessary to amend the statements of results such as the Löwenheim–Skolem theorem so that only normal models are considered.

First-order logic without equality is often employed in the context of second-order arithmetic and other higher-order theories of arithmetic, where the equality relation between sets of natural numbers is usually omitted.

Defining equality within a theory

If a theory has a binary formula $A(x,y)$ which satisfies reflexivity and Leibniz's law, the theory is said to have equality, or to be a theory with equality. The theory may not have all instances of the above schemas as axioms, but rather as derivable theorems. For example, in theories with no function symbols and a finite number of relations, it is possible to define equality in terms of the relations, by defining the two terms s and t to be equal if any relation is unchanged by changing s to t in any argument.

Some theories allow other *ad hoc* definitions of equality:

- In the theory of partial orders with one relation symbol $\leq$, one could define $s = t$ to be an abbreviation for $s \leq t \wedge t \leq s$.
- In set theory with one relation $\in$, one may define $s = t$ to be an abbreviation for $\forall x\,(s \in x \leftrightarrow t \in x) \wedge \forall x\,(x \in s \leftrightarrow x \in t)$. This definition of equality then automatically satisfies the axioms for equality. In this case, one should replace the usual axiom of extensionality, $\forall x \forall y [\forall z (z \in x \Leftrightarrow z \in y) \Rightarrow x = y]$, by $\forall x \forall y [\forall z (z \in x \Leftrightarrow z \in y) \Rightarrow \forall z (x \in z \Leftrightarrow y \in z)]$, i.e. if x and y have the same elements, then they belong to the same sets.

Metalogical properties

One motivation for the use of first-order logic, rather than higher-order logic, is that first-order logic has many metalogical properties that stronger logics do not have. These results concern general properties of first-order logic itself, rather than properties of individual theories. They provide fundamental tools for the construction of models of first-order theories.

Completeness and undecidability

Gödel's completeness theorem, proved by Kurt Gödel in 1929, establishes that there are sound, complete, effective deductive systems for first-order logic, and thus the first-order logical consequence relation is captured by finite provability. Naively, the statement that a formula φ logically implies a formula ψ depends on every model of φ; these models will in general be of arbitrarily large cardinality, and so logical consequence cannot be effectively verified by checking every model. However, it is possible to enumerate all finite derivations and search for a derivation of ψ from φ. If ψ is logically implied by φ, such a derivation will eventually be found. Thus first-order logical consequence is semidecidable: it is possible to make an effective enumeration of all pairs of sentences (φ,ψ) such that ψ is a logical consequence of φ.

Unlike propositional logic, first-order logic is undecidable (although semidecidable), provided that the language has at least one predicate of arity at least 2 (other than equality). This means that there is no decision procedure that determines whether arbitrary formulas are logically

valid. This result was established independently by Alonzo Church and Alan Turing in 1936 and 1937, respectively, giving a negative answer to the Entscheidungsproblem posed by David Hilbert in 1928. Their proofs demonstrate a connection between the unsolvability of the decision problem for first-order logic and the unsolvability of the halting problem.

There are systems weaker than full first-order logic for which the logical consequence relation is decidable. These include propositional logic and monadic predicate logic, which is first-order logic restricted to unary predicate symbols and no function symbols. Other logics with no function symbols which are decidable are the guarded fragment of first-order logic, as well as two-variable logic. The Bernays–Schönfinkel class of first-order formulas is also decidable. Decidable subsets of first-order logic are also studied in the framework of description logics.

The Löwenheim–Skolem theorem

The Löwenheim–Skolem theorem shows that if a first-order theory of cardinality λ has an infinite model, then it has models of every infinite cardinality greater than or equal to λ. One of the earliest results in model theory, it implies that it is not possible to characterize countability or uncountability in a first-order language. That is, there is no first-order formula $\varphi(x)$ such that an arbitrary structure M satisfies φ if and only if the domain of discourse of M is countable (or, in the second case, uncountable).

The Löwenheim–Skolem theorem implies that infinite structures cannot be categorically axiomatized in first-order logic. For example, there is no first-order theory whose only model is the real line: any first-order theory with an infinite model also has a model of cardinality larger than the continuum. Since the real line is infinite, any theory satisfied by the real line is also satisfied by some nonstandard models. When the Löwenheim–Skolem theorem is applied to first-order set theories, the nonintuitive consequences are known as Skolem's paradox.

The compactness theorem

The compactness theorem states that a set of first-order sentences has a model if and only if every finite subset of it has a model. This implies that if a formula is a logical consequence of an infinite set of first-order axioms, then it is a logical consequence of some finite number of those axioms. This theorem was proved first by Kurt Gödel as a consequence of the completeness theorem, but many additional proofs have been obtained over time. It is a central tool in model theory, providing a fundamental method for constructing models.

The compactness theorem has a limiting effect on which collections of first-order structures are elementary classes. For example, the compactness theorem implies that any theory that has arbitrarily large finite models has an infinite model. Thus the class of all finite graphs is not an elementary class (the same holds for many other algebraic structures).

There are also more subtle limitations of first-order logic that are implied by the compactness theorem. For example, in computer science, many situations can be modeled as a directed graph of states (nodes) and connections (directed edges). Validating such a system may require showing that no "bad" state can be reached from any "good" state. Thus one seeks to determine if the good and bad states are in different connected components of the graph. However, the compactness theorem can be used to show that connected graphs are not an elementary class in first-order logic, and there is no formula $\varphi(x,y)$ of first-order logic, in the logic of graphs, that expresses the idea that there is a path from x to y. Connectedness can be expressed in second-order logic, however, but not with only existential set quantifiers, as Σ_1^1 also enjoys compactness.

Lindström's theorem

Per Lindström showed that the metalogical properties just discussed actually characterize first-order logic in the sense that no stronger logic can also have those properties (Ebbinghaus and Flum 1994, Chapter XIII). Lindström defined a class of abstract logical systems, and a rigorous definition of the relative strength of a member of this class. He established two theorems for systems of this type:

- A logical system satisfying Lindström's definition that contains first-order logic and satisfies both the Löwenheim–Skolem theorem and the compactness theorem must be equivalent to first-order logic.
- A logical system satisfying Lindström's definition that has a semidecidable logical consequence relation and satisfies the Löwenheim–Skolem theorem must be equivalent to first-order logic.

Limitations

Although first-order logic is sufficient for formalizing much of mathematics, and is commonly used in computer science and other fields, it has certain limitations. These include limitations on its expressiveness and limitations of the fragments of natural languages that it can describe.

For instance, first-order logic is undecidable, meaning a sound, complete and terminating decision algorithm is impossible. This has led to the study of interesting decidable fragments such as C_2, first-order logic with two variables and the counting quantifiers $\exists^{\geq n}$ and $\exists^{\leq n}$ (these quantifiers are, respectively, "there exists at least n" and "there exists at most n") (Horrocks 2010).

Expressiveness

The Löwenheim–Skolem theorem shows that if a first-order theory has any infinite model, then it has infinite models of every cardinality. In particular, no first-order theory with an infinite model can be categorical. Thus there is no first-order theory whose only model has the set of natural numbers as its domain, or whose only model has the set of real numbers as its domain. Many extensions of first-order logic, including infinitary logics and higher-order logics, are more expressive in the sense that they do permit categorical axiomatizations of the natural numbers or real numbers. This expressiveness comes at a metalogical cost, however: by Lindström's theorem, the compactness theorem and the downward Löwenheim–Skolem theorem cannot hold in any logic stronger than first-order.

Formalizing natural languages

First-order logic is able to formalize many simple quantifier constructions in natural language, such as "every person who lives in Perth lives in Australia". But there are many more complicated features of natural language that cannot be expressed in (single-sorted) first-order logic. "Any logical system which is appropriate as an instrument for the analysis of natural language needs a much richer structure than first-order predicate logic" (Gamut 1991, p. 75).

Type	Example	Comment
Quantification over properties	If John is self-satisfied, then there is at least one thing he has in common with Peter	Requires a quantifier over predicates, which cannot be implemented in single-sorted first-order logic: $Zj \rightarrow \exists X(Xj \land Xp)$
Quantification over properties	Santa Claus has all the attributes of a sadist	Requires quantifiers over predicates, which cannot be implemented in single-sorted first-order logic: $\forall X(\forall x(Sx \rightarrow Xx) \rightarrow Xs)$
Predicate adverbial	John is walking quickly	Cannot be analysed as $Wj \land Qj$; predicate adverbials are not the same kind of thing as second-order predicates such as colour
Relative adjective	Jumbo is a small elephant	Cannot be analysed as $Sj \land Ej$; predicate adjectives are not the same kind of thing as second-order predicates such as colour
Predicate adverbial modifier	John is walking very quickly	-
Relative adjective modifier	Jumbo is terribly small	An expression such as "terribly", when applied to a relative adjective such as "small", results in a new composite relative adjective "terribly small"
Prepositions	Mary is sitting next to John	The preposition "next to" when applied to "John" results in the predicate adverbial "next to John"

Restrictions, extensions, and variations

There are many variations of first-order logic. Some of these are inessential in the sense that they merely change notation without affecting the semantics. Others change the expressive power more significantly, by extending the semantics through additional quantifiers or other new logical symbols. For example, infinitary logics permit formulas of infinite size, and modal logics add symbols for possibility and necessity.

Restricted languages

First-order logic can be studied in languages with fewer logical symbols than were described above.

- Because $\exists x \phi(x)$ can be expressed as $\neg \forall x \neg \phi(x)$, and $\forall x \phi(x)$ can be expressed as $\neg \exists x \neg \phi(x)$, either of the two quantifiers $\exists$ and $\forall$ can be dropped.
- Since $\phi \lor \psi$ can be expressed as $\neg(\neg \phi \land \neg \psi)$ and $\phi \land \psi$ can be expressed as $\neg(\neg \phi \lor \neg \psi)$, either $\lor$ or $\land$ can be dropped. In other words, it is sufficient to have $\neg$ and $\lor$, or $\neg$ and $\land$, as the only logical connectives.
- Similarly, it is sufficient to have only $\neg$ and $\rightarrow$ as logical connectives, or to have only the Sheffer stroke (NAND) or the Peirce arrow (NOR) operator.
- It is possible to entirely avoid function symbols and constant symbols, rewriting them via predicate symbols in an appropriate way. For example, instead of using a constant symbol 0 one may use a predicate $0(x)$ (interpreted as $x = 0$), and replace every predicate such as $P(0, y)$ with $\forall x \, (0(x) \rightarrow P(x, y))$. A function such as $f(x_1, x_2, \ldots, x_n)$ will similarly be replaced by a predicate $F(x_1, x_2, \ldots, x_n, y)$ interpreted as $y = f(x_1, x_2, \ldots, x_n)$. This change requires adding additional axioms to the theory at hand, so that interpretations of the predicate symbols used have the correct semantics.

Restrictions such as these are useful as a technique to reduce the number of inference rules or axiom schemas in deductive systems, which leads to shorter proofs of metalogical results. The cost of the restrictions is that it becomes more difficult to express natural-language statements in the formal system at hand, because the logical connectives used in the natural language statements must be replaced by their (longer) definitions in terms of the restricted collection of logical connectives. Similarly, derivations in the limited systems may be longer than derivations in systems that include additional connectives. There is thus a trade-off between the ease of working within the formal system and the ease of proving results about the formal system.

It is also possible to restrict the arities of function symbols and predicate symbols, in sufficiently expressive theories. One can in principle dispense entirely with functions of arity greater than 2 and predicates of arity greater than 1 in theories that include a pairing function. This is a function of arity 2 that takes pairs of elements of the domain and returns an ordered pair containing them. It is also sufficient to have two predicate symbols of arity 2 that define projection functions from an ordered pair to its components. In either case it is necessary that the natural axioms for a pairing function and its projections are satisfied.

Many-sorted logic

Ordinary first-order interpretations have a single domain of discourse over which all quantifiers range. **Many-sorted first-order logic** allows variables to have different **sorts**, which have different domains. This is also called **typed first-order logic**, and the sorts called **types** (as in data type), but it is not the same as first-order type theory. Many-sorted first-order logic is often used in the study of second-order arithmetic.

When there are only finitely many sorts in a theory, many-sorted first-order logic can be reduced to single-sorted first-order logic. One introduces into the single-sorted theory a unary predicate symbol for each sort in the many-sorted theory, and adds an axiom saying that these unary predicates partition the domain of discourse. For example, if there are two sorts, one adds predicate symbols $P_1(x)$ and $P_2(x)$ and the

axiom

$$\forall x(P_1(x) \lor P_2(x)) \land \neg \exists x(P_1(x) \land P_2(x)).$$

Then the elements satisfying P_1 are thought of as elements of the first sort, and elements satisfying P_2 as elements of the second sort. One can quantify over each sort by using the corresponding predicate symbol to limit the range of quantification. For example, to say there is an element of the first sort satisfying formula φ(x), one writes

$$\exists x(P_1(x) \land \phi(x)).$$

Additional quantifiers

Additional quantifiers can be added to first-order logic.

- Sometimes it is useful to say that "*P(x)* holds for exactly one *x*", which can be expressed as $\exists! x\, P(x)$. This notation, called uniqueness quantification, may be taken to abbreviate a formula such as $\exists x\, (P(x) \land \forall y\, (P(y) \to (x = y)))$.
- **First-order logic with extra quantifiers** has new quantifiers $Qx,...,$ with meanings such as "there are many *x* such that ...". Also see branching quantifiers and the plural quantifiers of George Boolos and others.
- **Bounded quantifiers** are often used in the study of set theory or arithmetic.

Infinitary logics

Infinitary logic allows infinitely long sentences. For example, one may allow a conjunction or disjunction of infinitely many formulas, or quantification over infinitely many variables. Infinitely long sentences arise in areas of mathematics including topology and model theory.

Infinitary logic generalizes first-order logic to allow formulas of infinite length. The most common way in which formulas can become infinite is through infinite conjunctions and disjunctions. However, it is also possible to admit generalized signatures in which function and relation symbols are allowed to have infinite arities, or in which quantifiers can bind infinitely many variables. Because an infinite formula cannot be represented by a finite string, it is necessary to choose some other representation of formulas; the usual representation in this context is a tree. Thus formulas are, essentially, identified with their parse trees, rather than with the strings being parsed.

The most commonly studied infinitary logics are denoted $L_{\alpha\beta}$, where α and β are each either cardinal numbers or the symbol ∞. In this notation, ordinary first-order logic is $L_{\omega\omega}$. In the logic $L_{\infty\omega}$, arbitrary conjunctions or disjunctions are allowed when building formulas, and there is an unlimited supply of variables. More generally, the logic that permits conjunctions or disjunctions with less than κ constituents is known as $L_{\kappa\omega}$. For example, $L_{\omega_1\omega}$ permits countable conjunctions and disjunctions.

The set of free variables in a formula of $L_{\kappa\omega}$ can have any cardinality strictly less than κ, yet only finitely many of them can be in the scope of any quantifier when a formula appears as a subformula of another.[9] In other infinitary logics, a subformula may be in the scope of infinitely many quantifiers. For example, in $L_{\kappa\infty}$, a single universal or existential quantifier may bind arbitrarily many variables simultaneously. Similarly, the logic $L_{\kappa\lambda}$ permits simultaneous quantification over fewer than λ variables, as well as conjunctions and disjunctions of size less than κ.

Non-classical and modal logics

- **Intuitionistic first-order logic** uses intuitionistic rather than classical propositional calculus; for example, ¬¬φ need not be equivalent to φ.
- First-order **modal logic** allows one to describe other possible worlds as well as this contingently true world which we inhabit. In some versions, the set of possible worlds varies depending on which possible world one inhabits. Modal logic has extra *modal operators* with meanings which can be characterized informally as, for example "it is necessary that φ" (true in all possible worlds) and "it is possible that φ" (true in some possible world). With standard first-order logic we have a single domain and each predicate is assigned one extension. With first-order modal logic we have a *domain function* that assigns each possible world its own domain, so that each predicate gets an extension only relative to these possible worlds. This allows us to model cases where, for example, Alex is a Philosopher, but might have been a Mathematician, and might not have existed at all. In the first possible world P(a) is true, in the second P(a) is false, and in the third possible world there is no *a* in the domain at all.
- **first-order fuzzy logics** are first-order extensions of propositional fuzzy logics rather than classical propositional calculus.

Fixpoint logic

Fixpoint logic extends first-order logic by adding the closure under the least fixed points of positive operators.[10]

Higher-order logics

The characteristic feature of first-order logic is that individuals can be quantified, but not predicates. Thus

$$\exists a(\text{Phil}(a))$$

is a legal first-order formula, but

$$\exists \text{Phil}(\text{Phil}(a))$$

is not, in most formalizations of first-order logic. Second-order logic extends first-order logic by adding the latter type of quantification. Other

higher-order logics allow quantification over even higher types than second-order logic permits. These higher types include relations between relations, functions from relations to relations between relations, and other higher-type objects. Thus the "first" in first-order logic describes the type of objects that can be quantified.

Unlike first-order logic, for which only one semantics is studied, there are several possible semantics for second-order logic. The most commonly employed semantics for second-order and higher-order logic is known as **full semantics**. The combination of additional quantifiers and the full semantics for these quantifiers makes higher-order logic stronger than first-order logic. In particular, the (semantic) logical consequence relation for second-order and higher-order logic is not semidecidable; there is no effective deduction system for second-order logic that is sound and complete under full semantics.

Second-order logic with full semantics is more expressive than first-order logic. For example, it is possible to create axiom systems in second-order logic that uniquely characterize the natural numbers and the real line. The cost of this expressiveness is that second-order and higher-order logics have fewer attractive metalogical properties than first-order logic. For example, the Löwenheim–Skolem theorem and compactness theorem of first-order logic become false when generalized to higher-order logics with full semantics.

Automated theorem proving and formal methods

Automated theorem proving refers to the development of computer programs that search and find derivations (formal proofs) of mathematical theorems. Finding derivations is a difficult task because the search space can be very large; an exhaustive search of every possible derivation is theoretically possible but computationally infeasible for many systems of interest in mathematics. Thus complicated heuristic functions are developed to attempt to find a derivation in less time than a blind search.

The related area of automated proof verification uses computer programs to check that human-created proofs are correct. Unlike complicated automated theorem provers, verification systems may be small enough that their correctness can be checked both by hand and through automated software verification. This validation of the proof verifier is needed to give confidence that any derivation labeled as "correct" is actually correct.

Some proof verifiers, such as Metamath, insist on having a complete derivation as input. Others, such as Mizar and Isabelle, take a well-formatted proof sketch (which may still be very long and detailed) and fill in the missing pieces by doing simple proof searches or applying known decision procedures: the resulting derivation is then verified by a small, core "kernel". Many such systems are primarily intended for interactive use by human mathematicians: these are known as proof assistants. They may also use formal logics that are stronger than first-order logic, such as type theory. Because a full derivation of any nontrivial result in a first-order deductive system will be extremely long for a human to write,[11] results are often formalized as a series of lemmas, for which derivations can be constructed separately.

Automated theorem provers are also used to implement formal verification in computer science. In this setting, theorem provers are used to verify the correctness of programs and of hardware such as processors with respect to a formal specification. Because such analysis is time-consuming and thus expensive, it is usually reserved for projects in which a malfunction would have grave human or financial consequences.

See also

- ACL2 — A Computational Logic for Applicative Common Lisp.
- Equiconsistency
- Extension by definitions
- Herbrandization
- Higher-order logic
- List of logic symbols
- Löwenheim number
- Prenex normal form
- Relational algebra
- Relational model
- Second-order logic
- Skolem normal form
- Tarski's World
- Truth table
- Type (model theory)
- Prolog

Notes

1. First Order Logic (http://people.sju.edu/~jhodgson/ugai/1order.html) CSC 2501: Symbolic Computation and AI, Dr. J.P.E. Hodgson, Saint Joseph's University, Philadelphia
2. Mendelson, Elliott (1964). *Introduction to Mathematical Logic*. Van Nostrand Reinhold. p. 56.
3. The word *language* is sometimes used as a synonym for signature, but this can be confusing because "language" can also refer to the set of formulas.
4. More precisely, there is only one language of each variant of one-sorted first-order logic: with or without equality, with or without functions, with or without propositional variables, ...
5. Some authors who use the term "well-formed formula" use "formula" to mean any string of symbols from the alphabet. However, most authors in mathematical logic use "formula" to mean "well-formed formula" and have no term for non-well-formed formulas. In every context, it is only the well-formed formulas that are of interest.
6. The SMT-LIB Standard: Version 2.0, by Clark Barrett, Aaron Stump, and Cesare Tinelli. http://smtlib.cs.uiowa.edu/language.shtml

7. e.g. the matrix shown at 4
8. e.g. the matrix shown at 2
9. Some authors only admit formulas with finitely many free variables in $L_{\kappa\omega}$, and more generally only formulas with $< \lambda$ free variables in $L_{\kappa\lambda}$.
10. Bosse, Uwe (1993). "An Ehrenfeucht–Fraïssé game for fixpoint logic and stratified fixpoint logic". In Börger, Egon. *Computer Science Logic: 6th Workshop, CSL'92, San Miniato, Italy, September 28 - October 2, 1992. Selected Papers*. Lecture Notes in Computer Science. **702**. Springer-Verlag. pp. 100–114. ISBN 3-540-56992-8. Zbl 0808.03024.
11. Avigad *et al.* (2007) discuss the process of formally verifying a proof of the prime number theorem. The formalized proof required approximately 30,000 lines of input to the Isabelle proof verifier.

References

- Andrews, Peter B. (2002); *An Introduction to Mathematical Logic and Type Theory: To Truth Through Proof*, 2nd ed., Berlin: Kluwer Academic Publishers. Available from Springer.
- Avigad, Jeremy; Donnelly, Kevin; Gray, David; and Raff, Paul (2007); "A formally verified proof of the prime number theorem", *ACM Transactions on Computational Logic*, vol. 9 no. 1 doi:10.1145/1297658.1297660 (https://dx.doi.org/10.1145%2F1297658.1297660)
- Barwise, Jon (1977); "An Introduction to First-Order Logic", in Barwise, Jon, ed. (1982). *Handbook of Mathematical Logic*. Studies in Logic and the Foundations of Mathematics. Amsterdam, NL: North-Holland. ISBN 978-0-444-86388-1.
- Barwise, Jon; and Etchemendy, John (2000); *Language Proof and Logic*, Stanford, CA: CSLI Publications (Distributed by the University of Chicago Press)
- Bocheński, Józef Maria (2007); *A Précis of Mathematical Logic*, Dordrecht, NL: D. Reidel, translated from the French and German editions by Otto Bird
- Ferreirós, José (2001); *The Road to Modern Logic — An Interpretation* (http://jstor.org/stable/2687794), Bulletin of Symbolic Logic, Volume 7, Issue 4, 2001, pp. 441–484, DOI 10.2307/2687794, JStor (http://links.jstor.org/sici?sici=1079-8986%28200112%297%3A4%3C441%3ATRTMLI%3E2.0.CO%3B2-O)
- Gamut, L. T. F. (1991); *Logic, Language, and Meaning, Volume 2: Intensional Logic and Logical Grammar*, Chicago, IL: University of Chicago Press, ISBN 0-226-28088-8
- Hilbert, David; and Ackermann, Wilhelm (1950); *Principles of Mathematical Logic*, Chelsea (English translation of *Grundzüge der theoretischen Logik*, 1928 German first edition)
- Hodges, Wilfrid (2001); "Classical Logic I: First Order Logic", in Goble, Lou (ed.); *The Blackwell Guide to Philosophical Logic*, Blackwell
- Ebbinghaus, Heinz-Dieter; Flum, Jörg; and Thomas, Wolfgang (1994); *Mathematical Logic*, Undergraduate Texts in Mathematics, Berlin, DE/New York, NY: Springer-Verlag, Second Edition, ISBN 978-0-387-94258-2
- Rautenberg, Wolfgang (2010), *A Concise Introduction to Mathematical Logic* (3rd ed.), New York, NY: Springer Science+Business Media, doi:10.1007/978-1-4419-1221-3, ISBN 978-1-4419-1220-6
- Tarski, Alfred and Givant, Steven (1987); *A Formalization of Set Theory without Variables*. Vol.41 of American Mathematical Society colloquium publications, Providence RI: American Mathematical Society, ISBN 978-0821810415.

External links

- Hazewinkel, Michiel, ed. (2001), "Predicate calculus", *Encyclopedia of Mathematics*, Springer, ISBN 978-1-55608-010-4
- Stanford Encyclopedia of Philosophy: Shapiro, Stewart; "Classical Logic (http://plato.stanford.edu/entries/logic-classical/)". Covers syntax, model theory, and metatheory for first-order logic in the natural deduction style.
- Magnus, P. D.; *forall x: an introduction to formal logic (http://www.fecundity.com/logic/)*. Covers formal semantics and proof theory for first-order logic.
- Metamath (http://us.metamath.org/index.html): an ongoing online project to reconstruct mathematics as a huge first-order theory, using first-order logic and the axiomatic set theory ZFC. *Principia Mathematica* modernized.
- Podnieks, Karl; *Introduction to mathematical logic (http://www.ltn.lv/~podnieks/)*
- *Cambridge Mathematics Tripos Notes* (http://john.fremlin.de/schoolwork/logic/index.html) (typeset by John Fremlin). These notes cover part of a past Cambridge Mathematics Tripos course taught to undergraduates students (usually) within their third year. The course is entitled "Logic, Computation and Set Theory" and covers Ordinals and cardinals, Posets and Zorn's Lemma, Propositional logic, Predicate logic, Set theory and Consistency issues related to ZFC and other set theories.
- Tree Proof Generator (http://www.umsu.de/logik/trees/) can validate or invalidate formulas of first-order logic through the semantic tableaux method.

Retrieved from "https://en.wikipedia.org/w/index.php?title=First-order_logic&oldid=731931219"

Categories: Systems of formal logic │ Predicate logic │ Model theory

---