Logic and Proof Hilary 2026

First-Order Logic

James Worrell

First-order logic can be understood as an extension of propositional logic. In propositional logic atomic formulas have no internal structure—they are propositional variables that are either true or false. In first-order logic atomic formulas are *predicates* that assert a relationship among certain elements. First-order logic also features *quantification*: the ability to assert that a certain property holds *for all elements* or that it holds *for some element*.

This lecture is mainly about definitions; we state results without proof. However, proofs (which are long but relatively straightforward inductions) can be found in Lecture 9 of the 23/24 course.

1 Syntax of First-Order Logic

The syntax of first-order logic is defined relative to a *signature*. A signature σ consists of a set of *constant symbols*, a set of *function symbols* and a set of *predicate symbols*. Each function and predicate symbol has an *arity* $k > 0$. We will often refer to predicates as *relations*. Typically we use letters c, d to denote constant symbols, f, g to denote function symbols and P, Q, R to denote predicate symbols. Note that the elements of a signature are *symbols*; later will will we interpret them as actual functions or relations. Independent of the signature σ we also have an infinite set of *variables* $x_0, x_1, x_2, \ldots$.

Given a signature σ, the set of *σ-terms* is defined by induction as follows:

- Each variable is a term.
- Each constant symbol is a term.
- If $t_1, \ldots, t_k$ are terms and f is a k-ary function symbol then $f(t_1, \ldots, t_k)$ is a term.

Next the set of *formulas* is defined inductively as follows:

- **true** and **false** are formulas.
- If $t_1, \ldots, t_k$ are terms and P is a k-ary predicate symbol, then $P(t_1, \ldots, t_k)$ is a formula.
- If F is a formula then $\neg F$ is a formula.
- If F, G are formulas then $(F \vee G)$ and $(F \wedge G)$ are both formulas.
- If F is a formula and x is a variable then $\exists x\, F$ and $\forall x\, F$ are both formulas.

Atomic formulas are those constructed according to the first two rules above. The atomic formula $P(t_1, \ldots, t_k)$ is read "$t_1, \ldots, t_k$ *are in relation* P". The symbol $\exists$ is called the *existential quantifier*. The formula $\exists x\, H$ is read "*there exists* x, H". The symbol $\forall$ is called the *universal quantifier*. The formula $\forall x\, H$ is read "*for all* x, H". A general first-order formula is built up from atomic formulas using the Boolean connectives and the two quantifiers. If a formula F occurs as part of another formula G then F is called a *subformula* of G. We assume that the quantifiers bind more tightly than any of the Boolean operators, e.g., $\forall x\, F \wedge G$ denotes the formula $(\forall x\, F) \wedge G$.

In a formula $\exists x\, G$ we say that G is the *scope* of the quantifier $\exists x$. The scope of an occurrence of the universal quantifier $\forall x$ is defined similarly. An occurrence of a variable x in a formula F is *bound* if that occurrence is within the scope of either $\exists x$ or $\forall x$. An occurrence that is not bound is said to be *free*. Note that the same variable can occur both bound and free in a given formula, e.g., variable x occurs both bound and free in the formula $P(x) \wedge \exists x\, P(x)$. A formula with no free variables is said to be *closed* or a *sentence*.

We will consider one important variant of first-order logic as described above, namely *first-order logic with equality*. This variant admits equality as built-in binary relation symbol. Thus, regardless of the signature, we admit $t_1 = t_2$ as an atomic formula for all terms t_1 and t_2.

2 Semantics of First-Order Logic

Given a signature σ, a *σ-structure* $\mathcal{A}$ consists of:

- a non-empty set A called the *universe* of the structure;
- for each k-ary predicate symbol P in σ, a k-ary relation $P^{\mathcal{A}} \subseteq A^k$;
- for each k-ary function symbol f in σ, a k-ary function, $f^{\mathcal{A}} : A^k \to A$;
- for each constant symbol c, an element $c^{\mathcal{A}}$ of A.

Given a structure $\mathcal{A}$ a *valuation* is a mapping v that assigns to each variable x an element $\mathsf{v}[\![x]\!] \in A$ of the universe. We denote by $\mathsf{v}_{[x \mapsto a]}$ the valuation such that $\mathsf{v}_{[x \mapsto a]}(x) = a$ and $\mathsf{v}_{[x \mapsto a]}(y) = \mathsf{v}(y)$ for all $y \neq x$. We extend v to a mapping on the set of σ-terms by induction as follows:

- for a constant symbol c we have $\mathsf{v}[\![c]\!] = c^{\mathcal{A}}$;
- for a k-ary function symbol f and terms $t_1, \ldots, t_k$ we have $\mathsf{v}[\![f(t_1, \ldots, t_k)]\!] = f^{\mathcal{A}}(\mathsf{v}[\![t_1]\!], \ldots, \mathsf{v}[\![t_k]\!])$.

We next define the *satisfaction relation* $\mathcal{A}, \mathsf{v} \models F$ by induction as follows:

1. $\mathcal{A}, \mathsf{v} \models \mathbf{true}$ and $\mathcal{A}, \mathsf{v} \not\models \mathbf{false}$;
2. $\mathcal{A}, \mathsf{v} \models P(t_1, \ldots, t_k)$ if and only if $(\mathsf{v}[\![t_1]\!], \ldots, \mathsf{v}[\![t_k]\!]) \in P^{\mathcal{A}}$;
3. $\mathcal{A}, \mathsf{v} \models (F \wedge G)$ if and only if $\mathcal{A}, \mathsf{v} \models F$ and $\mathcal{A}, \mathsf{v} \models G$;
4. $\mathcal{A}, \mathsf{v} \models (F \vee G)$ if and only if $\mathcal{A}, \mathsf{v} \models F$ or $\mathcal{A}, \mathsf{v} \models G$;
5. $\mathcal{A}, \mathsf{v} \models (\neg F)$ if and only if $\mathcal{A}, \mathsf{v} \not\models F$;
6. $\mathcal{A}, \mathsf{v} \models \exists x\, F$ if and only if there exists $a \in A$ such that $\mathcal{A}, \mathsf{v}_{[x \mapsto a]} \models G$;
7. $\mathcal{A}, \mathsf{v} \models \forall x\, F$ if and only if $\mathcal{A}, \mathsf{v}_{[x \mapsto a]} \models F$ for all $a \in A$.

If we are working in first-order logic with equality then we additionally have

7. If F is the formula $t_1 = t_2$ then $\mathcal{A}, \mathsf{v} \models F$ if and only if $\mathsf{v}[\![t_1]\!] = \mathsf{v}[\![t_2]\!]$.

While the function and predicate symbols in a signature can be interpreted as arbitrary functions and predicates in a given structure, the equality symbol is treated as a "built-in", and is always interpreted as equality.

In case $\mathcal{A}, \mathsf{v} \models F$ we say "*F is satisfied in structure $\mathcal{A}$ under valuation v*" or "*$\mathcal{A}, \mathsf{v}$ is a model of F*".

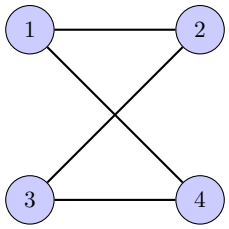

Figure 1: An undirected graph

Proposition 1. For a σ-formula F, structure $\mathcal{A}$, and valuations v, v' that agree on all free variables of F, we have $\mathcal{A}, \mathsf{v} \models F$ if and only if $\mathcal{A}, \mathsf{v}' \models F$.

Due to Proposition 1 we can talk about a structure $\mathcal{A}$ satisfying a sentence F (denoted $\mathcal{A} \models F$) without mentioning a valuation. More generally for a formula F whose free variables are included in the set $x_1, \ldots, x_n$, given $\boldsymbol{a} = (a_1, \ldots, a_n) \in A^n$ we write $\mathcal{A} \models F(\boldsymbol{a})$ to mean that $\mathcal{A}, \mathsf{v} \models F$ where v is a valuation that maps x_i to a_i for $i = 1, \ldots, n$.

Example 2. A undirected graph can be considered as a σ-structure for the signature σ with one binary relation symbol E, where E is interpreted as the edge relation. For example, the graph shown in Figure 2 can be represented by a structure $\mathcal{A}$ with universe $A = \{1, 2, 3, 4\}$ and irreflexive symmetric binary relation

$$E^{\mathcal{A}} = \{(1,2), (2,3), (3,4), (4,1), (2,1), (3,2), (4,3), (1,4)\}.$$

The following sentence asserts that edge relation is irreflexive and symmetric:

$$\forall x \, \neg E(x,x) \wedge \forall x \, \forall y \, (E(x,y) \to E(y,x))$$

This sentence is satisfied by the structure in Figure 2.

The following sentence expresses that every pair of nodes are connected by a path of length 3.

$$\forall x \, \forall y \, \exists z_1 \, \exists z_2 \, (E(x,z_1) \wedge E(z_1,z_2) \wedge E(z_2,y)).$$

This sentence is not satisfied by the structure in Figure 2.

Exercise 3. Let signature σ comprise a single unary relation symbol P and let $\mathcal{A}$ be the structure with universe $A = \{0, 1\}$ and $P^{\mathcal{A}} = \{1\}$. Does $\mathcal{A}$ satisfy the sentence

$$\forall x_1 \ldots \forall x_n (P(x_1) \to (P(x_2) \to (P(x_3) \to \ldots \to (P(x_n) \to P(x_1)) \ldots))) \, ?$$

Example 4. Consider a signature σ with one binary relation symbol $<$. A *totally ordered set* satisfies the following sentences in first-order logic with equality:

1. Irreflexivity: $\forall x \, \neg(x < x)$.
2. Transitivity: $\forall x \, \forall y \, \forall z ((x < y \wedge y < z) \to x < z)$.
3. Trichotomy: $\forall x \, \forall y \, (x < y \vee y < x \vee x = y)$.

The structures $(\mathbb{Z}, <_{\mathbb{Z}})$, $(\mathbb{Q}, <_{\mathbb{Q}})$ and $(\mathbb{R}, <_{\mathbb{R}})$ all satisfy the above sentences.

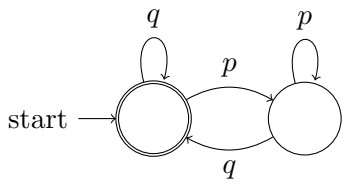

Figure 2: Automaton accepting all strings in which each q is followed by some p.

Example 5. Let $w = w_0 w_1 \dots w_{n-1}$ be a finite string over an alphabet $\{p, q\}$. Consider a signature σ with a binary relation symbol $<$ and unary predicate symbols P and Q. We can see w as a σ-structure $\mathcal{A}$ whose universe A is the set $\{0, 1, \dots, n-1\}$ of positions in w, $<^{\mathcal{A}}$ is the usual order on A, $P^{\mathcal{A}} = \{i : w_i = p\}$ is the set of positions in which letter a occurs and likewise $Q^{\mathcal{A}} = \{i : w_i = q\}$ is the set of positions in which letter b occurs.

The sentence $F = \forall x(P(x) \to \exists y\,(x < y \wedge Q(y)))$ is satisfied by a string precisely when every letter p is followed by a letter q. It is easy to see that the set of finite strings that satisfy F is precisely the language of the automaton in Figure 2. In fact for *any* sentence F over this signature, the set of strings satisfying F defines a regular language.

A first-order formula F over signature σ is *satisfiable* if $\mathcal{A}, \mathsf{v} \models F$ for some σ-structure $\mathcal{A}$ and valuation v. If F is not satisfiable it is called *unsatisfiable*. F is called *valid* if $\mathcal{A}, \mathsf{v} \models F$ for every σ-structure $\mathcal{A}$ and valuation v. Given a set of formulas $\boldsymbol{S}$ we write $\boldsymbol{S} \models F$ to mean that every σ-structure $\mathcal{A}$ and valuation v that satisfy every formula in $\boldsymbol{S}$ also satisfy F. The same relations exist among these notions as in propositional logic, e.g., F is unsatisfiable if and only if $\neg F$ is valid.

Exercise 6. Consider a signature σ with constant symbol 0, unary function symbol s, and unary predicate symbol P. Is the σ-formula $P(0) \wedge \forall x(P(x) \to P(s(x))) \wedge \exists x\, \neg P(x)$ satisfiable?

3 Standard Equivalences

For a given signature σ, two σ-formulas F and G are *logically equivalent* if for all σ-structures $\mathcal{A}$ and all valuations v we have $\mathcal{A}, \mathsf{v} \models F$ iff $\mathcal{A}, \mathsf{v} \models G$. In the rest of this section we summarise some useful equivalences.

Proposition 7. Let F and G be arbitrary σ-formulas. Then

(A) $\neg\forall x F \equiv \exists x \neg F$
$\neg\exists x F \equiv \forall x \neg F$

(B) If x does not occur free in G then:
$(\forall x F \wedge G) \equiv \forall x (F \wedge G)$ $\qquad$ $(\forall x F \vee G) \equiv \forall x (F \vee G)$
$(\exists x F \wedge G) \equiv \exists x (F \wedge G)$ $\qquad$ $(\exists x F \vee G) \equiv \exists x (F \vee G)$

A formula is in *prenex form* if it can be written

$$Q_1 y_1\, Q_2 y_2 \dots Q_n y_n\, F,$$

where $Q_i \in \{\exists, \forall\}$, $n \geq 0$, and F contains no quantifiers. We call F the **matrix** of the formula.

Example 8. We use Proposition 7 to transform the formula

$$\neg(\exists x\, P(x,y) \vee \forall z\, Q(z)) \wedge \exists w\, Q(w) \tag{1}$$

to prenex form by the following chain of equivalences:

$$\begin{aligned}
\neg(\exists x\, P(x,y) \vee \forall z\, Q(z)) \wedge \exists w\, Q(w) &\equiv (\neg\exists x\, P(x,y) \wedge \neg\forall z\, Q(z)) \wedge \exists w\, Q(w) \\
&\equiv (\forall x\, \neg P(x,y) \wedge \exists z\, \neg Q(z)) \wedge \exists w\, Q(w) \\
&\equiv \forall x\, \exists z\, (\neg P(x,y) \wedge \neg Q(z)) \wedge \exists w\, Q(w) \\
&\equiv \forall x\, \exists z\, \exists w\, ((\neg P(x,y) \wedge \neg Q(z)) \wedge Q(w))\,.
\end{aligned}$$

Note that in the above equational reasoning we use the fact that logical equivalence is a congruence with respect to the Boolean operators and quantifier (e.g., $F \equiv G$ implies $\forall x\, F \equiv \forall x\, G$, *etc.*)

Let F be a formula, x a variable, and t a term. Then $F[t/x]$ (read "F with t for x") denotes the formula with t substituted for every free occurrence of x in F. For example,

$$(\forall x\, P(x,y) \wedge Q(x))[t/x] = \forall x\, P(x,y) \wedge Q(t)\,.$$

Formally, we define $F[t/x]$ by induction on terms and formulas as follows. On terms we have:

$$\begin{aligned}
c[t/x] &= c \quad \text{for } c \text{ a constant symbol} \\
y[t/x] &= y \quad \text{for } y \neq x \text{ a variable} \\
x[t/x] &= t \\
f(t_1,\ldots,t_k)[t/x] &= f(t_1[t/x],\ldots,t_k[t/x]) \quad \text{for } f \text{ a } k\text{-ary function symbol}
\end{aligned}$$

We then extend the definition of $[t/x]$ to formulas as follows:

$$\begin{aligned}
P(t_1,\ldots,t_k)[t/x] &= P(t_1[t/x],\ldots,t_k[t/x]) \\
(\neg F)[t/x] &= \neg(F[t/x]) \\
(F \wedge G)[t/x] &= F[t/x] \wedge G[t/x] \\
(F \vee G)[t/x] &= F[t/x] \vee G[t/x] \\
(Qy\, F)[t/x] &= Qy\, (F[t/x]) \quad y \neq x \text{ a variable, } Q \in \{\forall,\exists\} \\
(Qx\, F)[t/x] &= Qx\, F \quad Q \in \{\forall,\exists\}\,.
\end{aligned}$$

A key fact about substitution is the following.

Lemma 9 (Translation Lemma)**.** Let t be a term and F a formula such that no variable in t occurs bound in F. Given a structure $\mathcal{A}$ and valuation v we have $\mathcal{A},\mathsf{v} \models F[t/x]$ iff $\mathcal{A},\mathsf{v}_{[x\mapsto \mathsf{v}[\![t]\!]]} \models F$.

To illustrate the necessity of the side-condition in the Translation Lemma, let F be the formula $\forall y P(x)$ and let $\mathcal{A}$ be the structure with universe $A = \{1,2\}$ and $P^{\mathcal{A}} = \{1\}$. Let v be the assignment such that $\mathsf{v}(x) = 2$. Then $F[y/x] = \forall y\, P(y)$ and so $\mathcal{A},\mathsf{v} \not\models F[y/x]$ but $\mathcal{A},\mathsf{v} \models \forall y, P(x)$. The Translation Lemma does not apply in this case because the variable y in the term to be substituted becomes bound by the quantifier $\forall y$ in F. This phenomenon is called *variable capture*.

We can rename bound variables in a formula while preserving logical equivalence. For example, we have $\forall x\, P(x) \equiv \forall y\, P(y)$. We make this idea formal in the following result, that can be proved using the Translation Lemma.

Proposition 10. Let $F = Qx\, G$ be a formula where $Q \in \{\forall,\exists\}$. Let y be a variable that does not occur in G. Then $F \equiv Qy\, (G[y/x])$.