Lecture Notes on Mathematical Logic

Vladimir Lifschitz

January 16, 2009

These notes provide an elementary, but mathematically solid, introduction to propositional and first-order logic. They contain many exercises.

Logic is the study of reasoning. The British mathematician and philosopher George Boole (1815–1864) is the man who made logic mathematical. His book *The Mathematical Analysis of Logic* was published in 1847.

Logic can be used in programming, and it can be applied to the analysis and automation of reasoning about software and hardware. This is why it is sometimes considered a part of theoretical computer science. Since reasoning plays an important role in intelligent behavior, logic is closely related to artificial intelligence.

The short book by the German philosopher Gottlob Frege (1848–1925) with the long title *Ideography, a Formula Language, Modeled upon that of Arithmetic, for Pure Thought* (1879), introduced notation that is somewhat similar to what is now called first-order formulas. Frege wrote:

> I believe that I can best make the relation of my ideography to ordinary language clear if I compare it to that which the microscope has to the eye. Because of the range of its possible uses and the versatility with which it can adapt to the most diverse circumstances, the eye is far superior to the microscope... But, as soon as scientific goals demand great sharpness of resolution, the eye proves to be insufficient.
>
> ... I am confident that my ideography can be successfully used wherever special value must be placed on the validity of proofs, as for example when the foundations of the differential and integral calculus are established.

In logic and in linguistics, we distinguish between two languages: the one that is the object of study and the one that we use to talk about that

object. The former is called the *object language*; the latter is the *metalanguage*. In Sections 1 and 2 below, the object language is the formal language of propositional formulas; in Section 3, we turn to first-order formulas. The metalanguage is the usual informal language of mathematics and theoretical computer science, which is a mixture of the English language and mathematical notation. The importance of distinguishing between the object language and the metalanguage was emphasized by the mathematician and logician Alfred Tarski (1902–1983), who taught logic at Berkeley since 1942.

1 Syntax and Semantics of Propositional Formulas

Propositional Formulas

A *propositional signature* is a set of symbols called *atoms*. (In examples, we will assume that p, q, r are atoms.) The symbols

$$\wedge \quad \vee \quad \rightarrow \quad \leftrightarrow \quad \neg \quad \bot \quad \top$$

are called *propositional connectives*. Among them, the symbols $\wedge$ *(conjunction)*, $\vee$ *(disjunction)*, $\rightarrow$ *(implication)* and $\leftrightarrow$ *(equivalence)* are called *2-place*, or *binary* connectives; $\neg$ *(negation)* is a *1-place*, or *unary* connective; $\bot$ *(false)* and $\top$ *(true)* are *0-place*.

Take a propositional signature σ which contains neither the propositional connectives nor the parentheses $(,)$. The alphabet of propositional logic consists of the atoms from σ, the propositional connectives, and the parentheses. By a *string* we understand a finite string of symbols in this alphabet. We define when a string is a *(propositional) formula* recursively, as follows:

- every atom is a formula,
- both 0-place connectives are formulas,
- if F is a formula then $\neg F$ is a formula,
- for any binary connective $\odot$, if F and G are formulas then $(F \odot G)$ is a formula.

For instance,

$$\neg(p \rightarrow q)$$

and

$$(\neg p \rightarrow q) \tag{1}$$

are formulas; the string

$$\neg p \to q \tag{2}$$

is not a formula. But very soon (see next page) we are going to introduce a convention according to which (2) can be used as an abbreviation for (1).

Properties of formulas can be often proved by induction. One useful method is strong induction on length (number of symbols). In such a proof, the induction hypothesis is that every formula which is shorter than F has the property P that we want to prove. From this assumption we need to derive that F has property P also. Then it follows that all formulas have property P.

In another useful form of induction, we check that all atoms and 0-place connectives have property P, and that the property is preserved when a new formula is formed using a unary or binary connective. More precisely, we show that

- every atom has property P,
- both 0-place connectives have property P,
- if a formula F has property P then so does $\neg F$,
- for any binary connective $\odot$, if formulas F and G have property P then so does $(F \odot G)$.

Then we can conclude that property P holds for all formulas. This is called "structural induction."

For instance, it is not difficult to prove, using either form of induction, that the number of left parentheses in any formula is equal to the number of right parentheses. (Do it!)

Problem 1.1 In any prefix of a formula, the number of left parentheses is greater than or equal to the number of right parentheses. (A *prefix* of a string $a_1 \cdots a_n$ is any string of the form $a_1 \cdots a_m$ where $0 \leq m \leq n$.)

Problem 1.2 Every prefix of a formula F

- is a string of negations (possibly empty), or
- has more left than right parentheses, or
- equals F.

Problem 1.3 No formula can be represented in the form $(F \odot G)$, where F and G are formulas and $\odot$ is a binary connective, in more than one way.

By representing a formula in the form $\neg F$ or $(F \odot G)$ we start "parsing" it. The assertion of the previous problem shows that a formula can be parsed in only one way.

From now on, we will abbreviate formulas of the form $(F \odot G)$ by dropping the outermost parentheses in them. We will also agree that $\leftrightarrow$ has a lower binding power than the other binary connectives. For instance,

$$p \vee q \leftrightarrow p \rightarrow r$$

will be viewed as shorthand for

$$((p \vee q) \leftrightarrow (p \rightarrow r)).$$

Finally, for any formulas $F_1, F_2, \ldots, F_n$ $(n > 2)$,

$$F_1 \wedge F_2 \wedge \cdots \wedge F_n$$

will stand for

$$(\cdots (F_1 \wedge F_2) \wedge \cdots \wedge F_n).$$

The abbreviation $F_1 \vee F_2 \vee \cdots \vee F_n$ will be understood in a similar way.

Semantics of Propositional Formulas

The symbols f and t are called *truth values*. An *interpretation* of a propositional signature σ is a function from σ into $\{\mathsf{f}, \mathsf{t}\}$. If σ is finite then an interpretation can be defined by the table of its values, for instance:

p	q	r
f	f	t

(3)

The semantics of propositional formulas that we are going to introduce defines which truth value is assigned to a formula F by an interpretation I.

As a preliminary step, we need to associate functions with all unary and binary connectives: a function from $\{\mathsf{f}, \mathsf{t}\}$ into $\{\mathsf{f}, \mathsf{t}\}$ with the unary connective $\neg$, and a function from $\{\mathsf{f}, \mathsf{t}\} \times \{\mathsf{f}, \mathsf{t}\}$ into $\{\mathsf{f}, \mathsf{t}\}$ with each of the binary connectives. These functions are denoted by the same symbols as the corresponding connectives, and defined by the following tables:

x	$\neg(x)$
f	t
t	f

x	y	$\wedge(x,y)$	$\vee(x,y)$	$\rightarrow(x,y)$	$\leftrightarrow(x,y)$
f	f	f	f	t	t
f	t	f	t	t	f
t	f	f	t	f	f
t	t	t	t	t	t

For any formula F and any interpretation I, the truth value F^I that is *assigned* to F by I is defined recursively, as follows:

- for any atom F, $F^I = I(F)$,
- $\bot^I = \mathsf{f}$, $\top^I = \mathsf{t}$,
- $(\neg F)^I = \neg(F^I)$,
- $(F \odot G)^I = \odot(F^I, G^I)$ for every binary connective $\odot$.

If $F^I = \mathsf{t}$ then we say that the interpretation I *satisfies* F (symbolically, $I \models F$).

Exercise: Find a formula F of the signature $\{p, q, r\}$ such that (3) is the only interpretation satisfying F.

If the underlying signature is finite then the set of interpretations is finite also, and the values of F^I for all interpretations I can be represented by a finite table. This table is called the *truth table* of F. For instance, the exercise above can be stated as follows: Find a formula with the truth table

p	q	r	F
f	f	f	f
f	f	t	t
f	t	f	f
f	t	t	f
t	f	f	f
t	f	t	f
t	t	f	f
t	t	t	f

Exercise: prove that for any formulas $F_1, \dots, F_n$ ($n \geq 1$) and any interpretation I,

$$(F_1 \wedge \cdots \wedge F_n)^I = \mathsf{t} \text{ iff } F_1^I = \cdots = F_n^I = \mathsf{t},$$
$$(F_1 \vee \cdots \vee F_n)^I = \mathsf{f} \text{ iff } F_1^I = \cdots = F_n^I = \mathsf{f}.$$

In the following two problems, we assume that the underlying signature is finite: $\sigma = \{p_1, \dots, p_n\}$.

Problem 1.4 For any interpretation I, there exists a formula F such that I is the only interpretation satisfying F.

Without the assumption that the underlying signature is finite, the assertion of Problem 1.4 would be incorrect. (Prove it!)

Problem 1.5 For any function α from interpretations to truth values, there exists a formula F such that, for all interpretations I, $F^I = \alpha(I)$.

Propositional formulas and truth tables were introduced in 1921 by the American mathematician and logician Emil Post (1897–1954). Post is known, along with Alan Turing, as one of the creators of theoretical computer science.

Tautologies

A propositional formula F is a *tautology* if every interpretation satisfies F. It is easy to check, for instance, that each of the formulas

$$(p \to q) \vee (q \to p),$$
$$((p \to q) \to p) \to p,$$
$$(p \to (q \to r)) \to ((p \to q) \to (p \to r))$$

is a tautology. (Do it!)

Equivalent Formulas

A formula F is *equivalent* to a formula G (symbolically, $F \sim G$) if, for every interpretation I, $F^I = G^I$. In other words, the metalanguage expression $F \sim G$ means that formula $F \leftrightarrow G$ is a tautology.

Here are several exercises related to the equivalence of propositional formulas.

(a) Conjunction and disjunction are associative:

$$(F \wedge G) \wedge H \sim F \wedge (G \wedge H),$$
$$(F \vee G) \vee H \sim F \vee (G \vee H).$$

Does equivalence have a similar property?

(b) Conjunction distributes over disjunction:

$$F \wedge (G \vee H) \sim (F \wedge G) \vee (F \wedge H);$$

disjunction distributes over conjunction:

$$F \vee (G \wedge H) \sim (F \vee G) \wedge (F \vee H).$$

Do these connectives distribute over equivalence?

(c) De Morgan's laws

$$\neg(F \wedge G) \sim \neg F \vee \neg G,$$
$$\neg(F \vee G) \sim \neg F \wedge \neg G$$

show how to transform a formula of the form $\neg(F \odot G)$ when $\odot$ is conjunction or disjunction. Find similar transformations for the cases when $\odot$ is implication or equivalence.

(d) Implication distributes over conjunction:

$$F \to (G \wedge H) \sim (F \to G) \wedge (F \to H).$$

Find a similar transformation for $(F \vee G) \to H$.

(e) To simplify a formula means to find an equivalent formula that is shorter. Simplify the following formulas:

(i) $F \leftrightarrow \neg F$,

(ii) $F \vee (F \wedge G)$,

(iii) $F \wedge (F \vee G)$,

(iv) $F \vee (\neg F \wedge G)$.

(v) $F \wedge (\neg F \vee G)$.

(f) For each of the formulas

$$p \wedge q,\ p \vee q,\ p \leftrightarrow q,\ \neg p, \top$$

find an equivalent formula that contains no connectives other than $\to$ and $\bot$.

(g) For each of the formulas

$$p \to q,\ p \wedge q$$

find an equivalent formula that contains no connectives other than $\leftrightarrow$ and $\vee$.

Adequate Sets of Connectives

Problem 1.6 For any formula, there exists an equivalent formula that contains no connectives other than $\rightarrow$ and $\bot$.

In this sense, $\{\rightarrow, \bot\}$ is an "adequate" set of connectives.
Similarly, we can check that then each of the sets

$$\{\wedge, \neg\},\ \{\vee, \neg\},\ \{\rightarrow, \neg\}$$

is adequate, under the assumption that the underlying signature is non-empty.

Problem 1.7 Any propositional formula equivalent to $\bot$ contains at least one of the connectives $\bot$, $\neg$.

This fact shows that the set $\{\wedge, \vee, \rightarrow, \leftrightarrow, \top\}$ is not adequate.

Disjunctive and Conjunctive Normal Forms

A *literal* is an atom or the negation of an atom. A *simple conjunction* is a formula of the form $L_1 \wedge \cdots \wedge L_n$ $(n \geq 1)$, where $L_1, \ldots, L_n$ are literals. A formula is in *disjunctive normal form* if it has the form $C_1 \vee \cdots \vee C_m$ $(m \geq 1)$, where $C_1, \ldots, C_m$ are simple conjunctions.

Problem 1.8 If the underlying signature is non-empty then any formula is equivalent to a formula in disjunctive normal form.

A *simple disjunction* is a formula of the form $L_1 \vee \cdots \vee L_n$ $(n \geq 1)$, where $L_1, \ldots, L_n$ are literals. A formula is in *conjunctive normal form* if it has the form $D_1 \wedge \cdots \wedge D_m$ $(m \geq 1)$, where $D_1, \ldots, D_m$ are simple disjunctions.

Problem 1.9 Let F be a formula in disjunctive normal form. Show that $\neg F$ is equivalent to a formula in conjunctive normal form.

Problem 1.10 If the underlying signature is non-empty then any formula is equivalent to a formula in conjunctive normal form.

Satisfiability and Entailment

A set Γ of formulas is *satisfiable* if there exists an interpretation that satisfies all formulas in Γ, and *unsatisfiable* otherwise. It is easy to check that a non-empty finite set $\{F_1, \ldots, F_n\}$ is unsatisfiable iff the conjunction $\neg(F_1 \wedge \cdots \wedge F_n)$ of its elements is a tautology. (Do it!)

Problem 1.11 Let Γ be a set of literals. Show that Γ is satisfiable iff there is no atom A for which both A and $\neg A$ belong to Γ.

For any atom A, the literals A, $\neg A$ are said to be *complementary* to each other. Thus the assertion of the last problem can be expressed as follows: A set of literals is satisfiable iff it does not contain complementary pairs.

A set Γ of formulas *entails* a formula F (symbolically, $\Gamma \models F$), if every interpretation that satisfies all formulas in Γ satisfies F also. Note that the notation for entailment uses the same symbol as the notation for satisfaction introduced earlier, the difference being that the expression on the left is an interpretation (I) in one case and a set of formulas (Γ) in the other. The formulas entailed by Γ are also called the *logical consequences* of Γ.

Exercise: Prove that $F_1, \ldots, F_n \models G$ iff $(F_1 \wedge \cdots \wedge F_n) \rightarrow G$ is a tautology. (In the first of these expressions, we dropped the braces $\{\,\}$ around $F_1, \ldots, F_n$.)

It is easy to check that for any set Γ of formulas and any formula F, $\Gamma \models F$ iff the set $\Gamma \cup \{\neg F\}$ is unsatisfiable. (Do it!)

2 Formal Proofs in Propositional Logic

In this section we study two deductive systems for propositional logic. Generally, a deductive system is a collection of axioms and inference rules. A "derivation" in a deductive system shows how a formula can be derived from a set of hypotheses using the postulates (axioms and inference rules) of the system.

Deductive System H

The first deductive system to be discussed here is a "Hilbert-style" system, and we will denote it by **H**. David Hilbert (1862–1943) was a German mathematician and logician, one of the most influential mathematicians of his time.

The derivable objects of **H** are propositional formulas that contain no connectives other than $\rightarrow$ and $\bot$, and in the discussion of **H** below all formulas are assumed to be in this class. (We know from Problem 1.6 that $\{\rightarrow, \bot\}$ is an adequate set of connectives.)

The set of axioms of **H** consists of the formulas of the following three forms:

H_1. $\quad F \rightarrow (G \rightarrow F)$.

H_2. $(F \to (G \to H)) \to ((F \to G) \to (F \to H))$.

H_3. $((F \to \bot) \to \bot) \to F$.

Each of the expressions H_1, H_2, H_3 is an "axiom schema" that represents infinitely many axioms of the system **H**, called the *instances* of the schema. For example, if p and q are atoms then the formula

$$(p \to q) \to ((\bot \to \bot) \to (p \to q))$$

is an instance of the axiom schema H_1.

The only inference rule of **H** is *modus ponens*:

$$\frac{F \quad F \to G}{G} .$$

The two expressions above the horizontal bar show the form of the *premises* to which the rule can be applied, and the expression under the bar shows the form of the *conclusion* that is derived by the rule. An *instance* of the rule is obtained from it by substituting formulas for the symbols F and G. For example,

$$\frac{p \quad p \to (q \to p)}{q \to p}$$

is an instance of *modus ponens* whose second premise is an instance of H_1.

Let Γ be a set of formulas, called *hypotheses*. In the system **H**, a *derivation* from Γ is a list $F_1, \ldots, F_n$ $(n \geq 1)$ of formulas such that, for every $i \leq n$, formula F_i

- is a hypothesis, or
- is an instance of one of the axiom schemas H_1, H_2, H_3, or
- for some $j, k < i$, can be derived from F_j and F_k by *modus ponens*.

A derivation whose last formula is F is said to be a *derivation of* F.

If there exists a derivation of F from Γ in the system **H** then we say that F is *derivable* from Γ in **H** and write $\Gamma \vdash_{\mathbf{H}} F$. A derivation from the empty set of hypotheses is called a *proof*. If there exists a proof of F in the system **H** then we say that F is *provable* in **H** and write $\vdash_{\mathbf{H}} F$. For other deductive systems, the terminology and notation are similar.

For instance, the following list of formulas is a derivation of p from $\bot$:

1.	$\bot$	— hypothesis
2.	$\bot \to ((p \to \bot) \to \bot)$	— H_1
3.	$(p \to \bot) \to \bot$	— from 1, 2
4.	$((p \to \bot) \to \bot) \to p$	— H_3
5.	p	— from 3, 4

The comments appended at the end of every line explain why this list of formulas is indeed a derivation. We have shown that

$$\bot \vdash_{\mathbf{H}} p. \tag{4}$$

Problem 2.1 $\vdash_{\mathbf{H}} p \to p$.

Problem 2.2 $\vdash_{\mathbf{H}} \bot \to p$.

Problem 2.3 For any set Γ of formulas and any formulas F and G, if $\Gamma \vdash_{\mathbf{H}} F$ and $\Gamma, F \vdash_{\mathbf{H}} G$ then $\Gamma \vdash_{\mathbf{H}} G$.

For instance, for any formula G, from

$$\bot \to p \vdash_{\mathbf{H}} G$$

we can conclude that $\vdash_{\mathbf{H}} G$.

Problem 2.4 For any set Γ of formulas and any formulas F and G, if $\Gamma, F \vdash_{\mathbf{H}} G$ then $\Gamma \vdash_{\mathbf{H}} F \to G$.

This fact is known as the *deduction theorem* for the system **H**. It allows us, for instance, to derive the assertion of Problem 2.2 from (4). The deduction theorem (for first-order logic) was first proved by the French mathematician Jacques Herbrand (1908–1931). The converse of the deduction theorem is true as well. (Prove it!)

As an exercise on the use of the deduction theorem, prove that

$$F \to \bot \vdash_{\mathbf{H}} F \to G.$$

Problem 2.5 $(F \to \bot) \to G \vdash_{\mathbf{H}} (G \to \bot) \to F$.

Problem 2.6 $F \to G, (F \to \bot) \to G \vdash_{\mathbf{H}} G$.

Soundness and Completeness of H

Problem 2.7 For any set Γ of formulas and any formula F, if $\Gamma \vdash_{\mathbf{H}} F$ then $\Gamma \models F$.

In particular, if $\vdash_{\mathbf{H}} F$ then F is a tautology. The assertion of Problem 2.7 expresses the *soundness* of deductive system **H**.

Problem 2.8 Let $A_1, \ldots, A_n$ be the atoms that occur in a formula F, and let I be an interpretation. Define the formulas $A'_1, \ldots, A'_n, F'$ as follows:

$$A'_i = \begin{cases} A_i, & \text{if } I \models A_i, \\ A_i \to \bot, & \text{otherwise;} \end{cases}$$

$$F' = \begin{cases} F, & \text{if } I \models F, \\ F \to \bot, & \text{otherwise.} \end{cases}$$

Show that $A'_1, \ldots, A'_n \vdash_{\mathbf{H}} F'$.

Problem 2.9 If F is a tautology then $\vdash_{\mathbf{H}} F$.

More generally, for any set Γ of formulas and any formula F, if $\Gamma \models F$ then $\Gamma \vdash_{\mathbf{H}} F$. This fact expresses the *completeness* of **H**. The completeness of a formal system for propositional logic was first proved by Emil Post in 1921.

The soundness and completeness theorems for **H** show that the relations $\models$ and $\vdash_{\mathbf{H}}$ are equivalent to each other.

Deductive System N

The second deductive system is the "natural deduction" system **N**.

A *sequent* is an expression of the form

$$\Gamma \Rightarrow F \tag{5}$$

("F under assumptions Γ") or

$$\Gamma \Rightarrow \tag{6}$$

("assumptions Γ are contradictory"), where Γ is a finite set of formulas. If Γ is written as $\{G_1, \ldots, G_n\}$, we will usually drop the braces and write (5) as

$$G_1, \ldots, G_n \Rightarrow F \tag{7}$$

and (6) as

$$G_1, \ldots, G_n \Rightarrow \ . \tag{8}$$

Intuitively, a sequent (7) is understood as the formula

$$(G_1 \wedge \cdots \wedge G_n) \to F \tag{9}$$

if $n > 0$, and as F if $n = 0$; (8) is understood as the formula

$$\neg(G_1 \wedge \cdots \wedge G_n) \tag{10}$$

if $n > 0$, and as $\bot$ if $n = 0$.

The derivable objects of **N** are sequents that contain no connectives other than

$$\wedge,\ \vee,\ \to,\ \neg\ .$$

The axioms of **N** are sequents of the forms

$$F \Rightarrow F$$

and

$$\Rightarrow F \vee \neg F.$$

The latter is called *the law of excluded middle*.

In the list of inference rules below, Γ, Δ, Δ_1, Δ_2 are finite sets of formulas, and Σ is a formula or the empty string. All inference rules of **N** except for the two rules at the end are classified into *introduction rules* (the left column) and *elimination rules* (the right column); the exceptions are the *contradiction rule* (C) and the *weakening rule* (W):

$$(\wedge I)\ \frac{\Gamma \Rightarrow F \quad \Delta \Rightarrow G}{\Gamma, \Delta \Rightarrow F \wedge G} \qquad (\wedge E)\ \frac{\Gamma \Rightarrow F \wedge G}{\Gamma \Rightarrow F} \quad \frac{\Gamma \Rightarrow F \wedge G}{\Gamma \Rightarrow G}$$

$$(\vee I)\ \frac{\Gamma \Rightarrow F}{\Gamma \Rightarrow F \vee G} \quad \frac{\Gamma \Rightarrow G}{\Gamma \Rightarrow F \vee G} \qquad (\vee E)\ \frac{\Gamma \Rightarrow F \vee G \quad \Delta_1, F \Rightarrow \Sigma \quad \Delta_2, G \Rightarrow \Sigma}{\Gamma, \Delta_1, \Delta_2 \Rightarrow \Sigma}$$

$$(\rightarrow I)\ \frac{\Gamma, F \Rightarrow G}{\Gamma \Rightarrow F \rightarrow G} \qquad (\rightarrow E)\ \frac{\Gamma \Rightarrow F \quad \Delta \Rightarrow F \rightarrow G}{\Gamma, \Delta \Rightarrow G}$$

$$(\neg I)\ \frac{\Gamma, F \Rightarrow}{\Gamma \Rightarrow \neg F} \qquad (\neg E)\ \frac{\Gamma \Rightarrow F \quad \Delta \Rightarrow \neg F}{\Gamma, \Delta \Rightarrow}$$

$$(C)\ \frac{\Gamma \Rightarrow}{\Gamma \Rightarrow F}$$

$$(W)\ \frac{\Gamma \Rightarrow \Sigma}{\Gamma, \Delta \Rightarrow \Sigma}$$

System **N** is sound and complete:

(a) a sequent $\Gamma \Rightarrow F$ is provable in **N** iff Γ entails F;

(b) a sequent $\Gamma \Rightarrow$ is provable in **N** iff Γ is unsatisfiable.

To prove a formula F in System **N** means to prove the sequent $\Rightarrow F$. Assertion (a) above shows that a formula is provable in **N** iff it is a tautology.

Natural deduction was invented by the German logician Gerhard Gentzen (1909–1945).

A_1.	$p \to (q \to r)$		
1.	$A_1 \Rightarrow p \to (q \to r)$	— axiom	Assume $p \to (q \to r)$. Now our goal is to prove $(p \wedge q) \to r$.
A_2.	$p \wedge q$		
2.	$A_2 \Rightarrow p \wedge q$	— axiom	Assume $p \wedge q$. Now our goal is to prove r.
3.	$A_2 \Rightarrow p$	— by $(\wedge E)$ from 2	From the second assumption, p.
4.	$A_1, A_2 \Rightarrow q \to r$	— by $(\to E)$ from 3, 1	Consequently, by the first assumption, $q \to r$.
5.	$A_2 \Rightarrow q$	— by $(\wedge E)$ from 2	On the other hand, from the second assumption, q.
6.	$A_1, A_2 \Rightarrow r$	— by $(\to E)$ from 5, 4	Consequently r.
7.	$A_1 \Rightarrow (p \wedge q) \to r$	— by $(\to I)$ from 6	Thus we proved $(p \wedge q) \to r$
8.	$\Rightarrow (p \to (q \to r)) \to ((p \wedge q) \to r)$	— by $(\to I)$ from 7	and consequently we are done.

Figure 1: A proof of $(p \to (q \to r)) \to ((p \wedge q) \to r)$

In proofs in system **N**, it is convenient to denote the "assumptions" appearing in sequents to the left of $\Rightarrow$ by $A_1, A_2, \ldots$. Figure 1 shows a proof of the formula

$$(p \to (q \to r)) \to ((p \wedge q) \to r)$$

written in this manner, with the corresponding "informal proof" to the right of the bar. Figure 2 is a proof of

$$(\neg p \to q) \to (\neg q \to p)$$

in **N**, again with an "English translation."

As an exercise, find a proof of each of the formulas

$$(p \to (q \to r)) \to ((p \to q) \to (p \to r)),$$
$$(p \to q) \to (\neg q \to \neg p)$$

in System **N**, and show also the corresponding informal proof.

A_1.	$\neg p \to q$		
1.	$A_1 \Rightarrow \neg p \to q$	— axiom	Assume $\neg p \to q$. Now our goal is to prove $\neg q \to p$.
A_2.	$\neg q$		
2.	$A_2 \Rightarrow \neg q$	— axiom	Assume $\neg q$. Now our goal is to prove p.
3.	$\Rightarrow p \vee \neg p$	— axiom	Consider two cases.
A_3.	p		
4.	$A_3 \Rightarrow p$	— axiom	Case 1: p. This case is trivial.
A_4.	$\neg p$		
5.	$A_4 \Rightarrow \neg p$	— axiom	Case 2: $\neg p$.
6.	$A_1, A_4 \Rightarrow q$	— by $(\to E)$ from 5, 1	Then, by the first assumption, q.
7.	$A_1, A_2, A_4 \Rightarrow$	— by $(\neg E)$ from 6, 2	This contradicts the second assumption,
8.	$A_1, A_2, A_4 \Rightarrow p$	— by (C) from 7	so that we can conclude p also.
9.	$A_1, A_2 \Rightarrow p$	— by $(\vee E)$ from 3, 4, 8	Thus, in either case, p.
10.	$A_1 \Rightarrow \neg q \to p$	— by $(\to I)$ from 9	We have proved $\neg q \to p$
11.	$\Rightarrow (\neg p \to q) \to (\neg q \to p)$	— by $(\to I)$ from 10	and consequently we are done.

Figure 2: A proof of $(\neg p \to q) \to (\neg q \to p)$

Do the same for the formulas in the following problems:

Problem 2.10 $((p \wedge q) \vee r) \to (p \vee r)$.

Problem 2.11 $(p \to q) \vee (q \to p)$.

3 First-Order Logic

Predicate Formulas

"Predicate formulas" generalize the concept of a propositional formula defined in Section 1.

A *predicate signature* is a set of symbols of two kinds—*object constants* and *predicate constants*—with a nonnegative integer, called the *arity*, assigned to every predicate constant. A predicate constant is said to be *propositional* if its arity is 0. Propositional constants are similar to atoms in propositional logic. A predicate constant is *unary* if its arity is 1, and *binary* if its arity is 2. For instance, we can define a predicate signature

$$\{a, P, Q\} \tag{11}$$

by saying that a is an object constant, P is a unary predicate constant, and Q is a binary predicate constant.

Take a predicate signature σ that does not include any of the following symbols:

- the *object variables* $x, y, z, x_1, y_1, z_1, x_2, y_2, z_2, \ldots$,
- the propositional connectives,
- the *universal quantifier* $\forall$ and the *existential quantifier* $\exists$,
- the parentheses and the comma.

The alphabet of predicate logic consists of the elements of σ and of the four groups of additional symbols listed above. A *string* is a finite string of symbols in this alphabet.

A *term* is an object constant or an object variable. A string is called an *atomic formula* if it is a propositional constant or has the form

$$R(t_1, \ldots, t_n)$$

where R is a predicate constant of arity n ($n > 0$) and $t_1, \ldots, t_n$ are terms. For instance, if the underlying signature is (11) then $P(a)$ and $Q(a, x)$ are atomic formulas.

We define when a string is a *(predicate) formula* recursively, as follows:

- every atomic formula is a formula,
- both 0-place connectives are formulas,
- if F is a formula then $\neg F$ is a formula,
- for any binary connective $\odot$, if F and G are formulas then $(F \odot G)$ is a formula,

- for any quantifier K and any variable v, if F is a formula then KvF is a formula.

For instance, if the underlying signature is (11) then

$$(\neg P(a) \vee \exists x(P(x) \wedge Q(x, y)))$$

is a formula.

When we write predicate formulas, we will drop some parentheses and use other abbreviations introduced in Section 1. A string of the form $\forall v_1 \cdots \forall v_n$ $(n \geq 0)$ will be written as $\forall v_1 \cdots v_n$, and similarly for the existential quantifier.

An occurrence of a variable v in a formula F is *bound* if it occurs in a part of F of the form KvG; otherwise it is *free* in F. For instance, in the formula

$$\exists y P(x, y) \wedge \neg \exists x P(x, x) \tag{12}$$

the first occurrence of x is free, and the other three are bound; both occurrences of y in (12) are bound. We say that v is *free (bound)* in F if some occurrence of v is free (bound) in F. A formula without free variables is called a *closed formula*, or a *sentence*.

Representing English Sentences by Predicate Formulas

Before we continue the study of the syntax of predicate logic, it is useful to get some experience in translating sentences from English into the language of predicate formulas. The translation exercises below are different from the other problems in that they are not precisely stated mathematical questions: there is no way to prove, in the mathematical sense, that a translation is adequate (at least until the semantics of predicate logic is defined). There is sometimes no clear-cut difference between good and bad translations; it may happen that one translation is somewhat less adequate than another but still satisfactory.

In these translation exercises, the underlying signature is (11). We will think of object variables as ranging over the set ω of nonnegative integers, and interpret the signature as follows:

- a represents the number 10,
- $P(x)$ represents the condition "x is a prime number,"
- $Q(x, y)$ represents the condition "x is less than y."

As an example, the sentence *All prime numbers are greater than* x can be represented by the formula

$$\forall y(P(y) \rightarrow Q(x, y)). \tag{13}$$

In the following two problems, represent the given English sentences by predicate formulas.

Problem 3.1 (a) There is a prime number that is less than 10. (b) x equals 0. (c) x equals 9.

Problem 3.2 There are infinitely many prime numbers.

Substitution

Let F be a formula and v a variable. The result of the *substitution* of a term t for v in F is the formula obtained from F by replacing each free occurrence of v by t. When we intend to consider substitutions for v in a formula, it is convenient to denote this formula by an expression like $F(v)$; then we can denote the result of the substitution of a term t for v in this formula by $F(t)$. For instance, if we denote formula (12) by $F(x)$ then $F(a)$ stands for

$$\exists y P(a, y) \wedge \neg \exists x P(x, x).$$

Let $F(x)$ be formula (13) proposed above as a translation for the condition "all prime numbers are greater than x." A formula of the form $F(t)$, where t is a term, *usually* expresses the same condition applied to the value of t. For instance, $F(a)$ is

$$\forall y(P(y) \rightarrow Q(a, y)),$$

which means that all prime numbers are greater than 10; $F(z_2)$ is

$$\forall y(P(y) \rightarrow Q(z_2, y)),$$

which means that all prime numbers are greater than z_2. There is one exception, however. The formula $F(y)$, that is,

$$\forall y(P(y) \rightarrow Q(y, y)),$$

expresses the (incorrect) assertion "every prime number is less than itself." The problem with this substitution is that y, when substituted for x in $F(x)$, is "captured" by a quantifier. To express the assertion "all prime numbers

are greater than y" by a predicate formula, we would have to use a bound variable different from y and write, for instance,

$$\forall z(P(z) \to Q(y, z)).$$

To distinguish "bad" substitutions, as in the last example, from "good" substitutions, we introduce the following definition. A term t is *substitutable* for a variable v in a formula F if

- t is a constant, or
- t is a variable w, and no part of F of the form KwG contains an occurrence of v which is free in F.

For instance, a and z_2 are substitutable in (13) for x, and y is not substitutable.

Semantics of Predicate Formulas

The semantics of propositional formulas described in Section 1 defined which truth value F^I is assigned to a propositional formula F by an interpretation I. Our next goal is to extend this definition to predicate formulas. First we need to extend the definition of an interpretation to predicate signatures.

An *interpretation* I of a predicate signature σ consists of

- a non-empty set $|I|$, called the *universe* of I,
- for every object constant c of σ, an element c^I of $|I|$,
- for every propositional constant R of σ, an element R^I of $\{\mathsf{f}, \mathsf{t}\}$,
- for every predicate constant R of σ of arity $n > 0$, a function R^I from $|I|^n$ to $\{\mathsf{f}, \mathsf{t}\}$.

For instance, the second paragraph of the section on representing English sentences by predicate formulas can be viewed as the definition of an interpretation of signature (11). For this interpretation I,

$$|I| = \omega,$$

$$a^I = 10,$$

$$P^I(n) = \begin{cases} \mathsf{t}, & \text{if } n \text{ is prime,} \\ \mathsf{f}, & \text{otherwise,} \end{cases} \tag{14}$$

$$Q^I(m, n) = \begin{cases} \mathsf{t}, & \text{if } m < n, \\ \mathsf{f}, & \text{otherwise.} \end{cases}$$

The semantics of predicate logic introduced below defines the truth value F^I only for the case when F is a *sentence*. As in propositional logic, the definition is recursive. For propositional constants, there is nothing to define: the truth value R^I is part of the interpretation I. For other atomic sentences, we can define

$$R(t_1, \ldots, t_n)^I = R^I(t_1^I, \ldots, t_n^I).$$

(Since $R(t_1, \ldots, t_n)$ is a sentence, each term t_i is an object *constant*, and consequently t_i^I is part of I). For propositional connectives, we can use the same clauses as in propositional logic. But the case of quantifiers presents a difficulty. One possibility would be to define

$$\exists w F(w)^I = \mathsf{t} \text{ iff, for some object constant } c,\ F(c)^I = \mathsf{t}$$

and similarly for the universal quantifier. But this formulation is unsatisfactory: it disregards the fact that some elements of the universe $|I|$ may be not represented by object constants. For instance, in the example above 10 is the only element of the universe ω for which there is a corresponding object constant in signature (11); we expect to find that

$$(\exists x Q(x, a))^I = \mathsf{t}$$

(there exists a number that is less than 10) although

$$Q(a, a)^I = \mathsf{f}$$

(10 does not have this property). Because of this difficulty, some additional work is needed before we define F^I.

Consider an interpretation I of a predicate signature σ. For any element ξ of its universe $|I|$, select a new symbol ξ^*, called the *name* of ξ. By σ^I we denote the predicate signature obtained from σ by adding all names ξ^* as additional object constants. For instance, if σ is (11) and I is (14) then

$$\sigma^I = \{a, 0^*, 1^*, 2^*, \ldots, P, Q\}.$$

The interpretation I can be extended to the new signature σ^I by defining

$$(\xi^*)^I = \xi$$

for all $\xi \in |I|$. We will denote this interpretation of σ^I by the same symbol I.

We will define recursively the truth value F^I that is *assigned* to F by I for every sentence F of the extended signature σ^I; that includes, in particular, every sentence of the signature σ. For any propositional constant R, R^I is part of the interpretation I. Otherwise, we define:

- $R(t_1, \dots, t_n)^I = R^I(t_1^I, \dots, t_n^I)$,
- $\bot^I = \mathsf{f}$, $\top^I = \mathsf{t}$,
- $(\neg F)^I = \neg(F^I)$,
- $(F \odot G)^I = \odot(F^I, G^I)$ for every binary connective $\odot$,
- $\forall w F(w)^I = \mathsf{t}$ iff, for all $\xi \in |I|$, $F(\xi^*)^I = \mathsf{t}$,
- $\exists w F(w)^I = \mathsf{t}$ iff, for some $\xi \in |I|$, $F(\xi^*)^I = \mathsf{t}$.

As in propositional logic, we say that I *satisfies* F, and write $I \models F$, if $F^I = \mathsf{t}$.

Problem 3.3 Show that interpretation (14) satisfies $\exists x Q(x, a)$.

Problem 3.4 Let $F(x)$ be formula (13). Show that, for every $n \in \omega$, (14) satisfies $F(n^*)$ iff $n < 2$.

Satisfiability

If there exists an interpretation satisfying a sentence F, we say that F is *satisfiable*. A set Γ of sentences is *satisfiable* if there exists an interpretation that satisfies all sentences in Γ.

In each of the following problems, determine whether the given set of sentences is satisfiable. We assume that P and Q are predicate constants; their arities will be every time clear from the context.

Problem 3.5 (a) $P(a)$, $\exists x \neg P(x)$. (b) $P(a)$, $\forall x \neg P(x)$.

Problem 3.6 $\forall x \exists y P(x, y)$, $\forall x \neg P(x, x)$.

Entailment

A set Γ of sentences *entails* a sentence F, or is a *logical consequence* of Γ (symbolically, $\Gamma \models F$), if every interpretation that satisfies all sentences in Γ satisfies F.

As an exercise, determine whether the sentences

$$\exists x P(x),\ \exists x Q(x)$$

entail

$$\exists x(P(x) \wedge Q(x)).$$

A sentence F is *logically valid* if every interpretation satisfies F. This concept is similar to the notion of a tautology.

The *universal closure* of a formula F is the sentence $\forall v_1 \cdots v_n F$, where $v_1, \ldots, v_n$ are all free variables of F. About a formula with free variables we say that it is *logically valid* if its universal closure is logically valid. A formula F is *equivalent* to a formula G (symbolically, $F \sim G$) if the formula $F \leftrightarrow G$ is logically valid.

Problem 3.7 For each of the formulas

$$P(x) \rightarrow \exists x P(x),$$
$$P(x) \rightarrow \forall x P(x)$$

determine whether it is logically valid.

Function Symbols and Equality: Syntax

Predicate logic as defined at the beginning of this section is somewhat more restricted than what is usually called "first-order logic," and now our goal is to remove these restrictions. First, we will generalize the notion of a term. In addition to object constants and object variables, we will allow terms to be formed using symbols for functions, "function constants." Second, we will add the equal sign to the language, and equalities will be included as a new kind of atomic formulas.

Our most general notion of a signature is defined as follows. A *signature* is a set of symbols of two kinds—*function constants* and *predicate constants*—with a nonnegative integer, called the *arity*, assigned to each symbol. An *object constant* is a function constant of arity 0. A function constant is *unary* if its arity is 1, and *binary* if its arity is 2. *Propositional constants*, as well as *unary* and *binary* predicate constants, are defined as above. *Terms* are defined recursively, as follows:

- every object constant is a term,
- every object variable is a term,
- for every function constant h of arity n ($n > 0$), if $t_1, \ldots, t_n$ are terms then so is $h(t_1, \ldots, t_n)$.

In first-order logic, there are three kinds of *atomic formulas*: propositional constants, strings of the form $R(t_1, \ldots, t_n)$ where R is a predicate constant of arity n ($n > 0$) and $t_1, \ldots, t_n$ are terms, and strings of the form $(t_1 = t_2)$ where t_1, t_2 are terms. Given this set of atomic formulas, the definition of *(first-order) formulas* is the same as the definition of predicate formulas at the beginning of this section.

For any terms t_1 and t_2, $t_1 \neq t_2$ stands for the formula $\neg(t_1 = t_2)$.

The definitions of *free* and *bound* occurrences of a variable in a formula remain the same as before. (Note that these concepts apply to a formula, but not to a term. Sometimes, all variables occurring in a term are considered its "free variables.") The *substitution* of a term for a variable is also defined as before. The definition of a substitutable term, in the presence of function symbols, is stated as follows: A term t is *substitutable* for a variable v in a formula F if, for each variable w occurring in t, no part of F of the form KwG contains an occurrence of v which is free in F.

Function Symbols and Equality: Semantics

To generalize the notion of an interpretation to arbitrary signatures, we need to say how one is allowed to interpret a function constant of arity $n > 0$. Such a symbol can be interpreted as any total function of n variables whose arguments and values come from the universe of the interpretation. Thus an *interpretation* I of a signature σ consists of

- a non-empty set $|I|$, called the *universe* of I,
- for every object constant c of σ, an element c^I of $|I|$,
- for every function constant h of σ of arity $n > 0$, a function h^I from $|I|^n$ to $|I|$,
- for every propositional constant R of σ, an element R^I of $\{\mathsf{f}, \mathsf{t}\}$,
- for every predicate constant R of σ of arity $n > 0$, a function R^I from $|I|^n$ to $\{\mathsf{f}, \mathsf{t}\}$.

As an example, consider the *signature of first-order arithmetic*

$$\{a, s, f, g\}, \tag{15}$$

where a is an object constant (intended to represent 0), s is a unary function constant (for the successor function), and f, g are binary function constants (for addition and multiplication). Since this signature includes no predicate

constants, its only atomic formulas are equalities. The intended interpretation I of (15) is defined as follows:

$$\begin{array}{l} |I| = \omega, \\ a^I = 0, \\ s^I(n) = n + 1, \\ f^I(m, n) = m + n, \\ g^I(m, n) = m \cdot n. \end{array} \tag{16}$$

Problem 3.8 Represent the following English sentences by first-order formulas:

- There exists at most one x such that $P(x)$.
- There exists exactly one x such that $P(x)$.
- There exist at least two x such that $P(x)$.
- There exist at most two x such that $P(x)$.
- There exist exactly two x such that $P(x)$.

Now let us go back to the semantics of first-order logic. For the definition of F^I to be applicable in the presence of function constants of arity > 0, we need to extend the notation t^I from object constants to arbitrary variable-free terms of the signature σ^I, and to add a clause for equality. The value t^I *assigned* by I to t is defined recursively by the formula

$$h(t_1, \ldots, t_n)^I = h^I(t_1^I, \ldots, t_n^I).$$

The additional clause in the definition of satisfaction reads as follows:

- $(t_1 = t_2)^I = \mathsf{t}$ iff $t_1^I = t_2^I$.

The definitions of satisfaction and of all other semantic concepts introduced above for formulas of a predicate signature apply to formulas with function constants and equality without any changes.

Problem 3.9 For each of the following sentences determine whether it is satisfiable:

- $a = b$,
- $\forall xy(x = y)$,
- $\forall xy(x \neq y)$.

Natural Deduction in First-Order Logic

In the first-order version of deductive system **N**, the derivable objects are sequents

$$\Gamma \Rightarrow F \quad \text{and} \quad \Gamma \Rightarrow$$

where F and the elements of Γ are formed using the connectives $\wedge, \vee, \rightarrow, \neg$ and the quantifiers $\forall, \exists$. In addition to the axiom schemas of propositional **N** (Section 2), the first-order version includes

$$\Rightarrow t = t$$

where t is a term. The additional inference rules of the system are:

$$(\forall I)\ \frac{\Gamma \Rightarrow F(v)}{\Gamma \Rightarrow \forall v F(v)} \qquad (\forall E)\ \frac{\Gamma \Rightarrow \forall v F(v)}{\Gamma \Rightarrow F(t)}$$

where v is not a free variable of any formula in Γ

where t is substitutable for v in $F(v)$

$$(\exists I)\ \frac{\Gamma \Rightarrow F(t)}{\Gamma \Rightarrow \exists v F(v)} \qquad (\exists E)\ \frac{\Gamma \Rightarrow \exists v F(v) \quad \Delta, F(v) \Rightarrow \Sigma}{\Gamma, \Delta \Rightarrow \Sigma}$$

where t is substitutable for v in $F(v)$

where v is not a free variable of any formula in Δ, Σ

$$(\mathit{Repl})\ \frac{\Gamma \Rightarrow t_1 = t_2 \quad \Delta \Rightarrow F(t_1)}{\Gamma, \Delta \Rightarrow F(t_2)} \qquad \frac{\Gamma \Rightarrow t_1 = t_2 \quad \Delta \Rightarrow F(t_2)}{\Gamma, \Delta \Rightarrow F(t_1)}$$

where t_1 and t_2 are substitutable for v in $F(v)$.

This formal system is sound and complete: if a sequent can be proved using the axioms and inference rules shown above then it is logically valid, and the other way around. The completeness of a formal system for first-order logic was first proved by Kurt Gödel (1906–1978) in 1930. Gödel was a member of the Institute for Advanced Study since 1940. His other well-known result, Gödel's incompleteness theorem, is related to the problem of formalizing the arithmetic of natural numbers, and it is not discussed in these notes.

In each of the following problems, prove the given formula in System **N**. (If the formula is an equivalence, rewrite it first as a conjunction of two implications.)

Problem 3.10 $(P(a) \wedge \forall x(P(x) \rightarrow Q(x))) \rightarrow Q(a)$.

Problem 3.11 $P(a) \rightarrow \neg\forall x\neg P(x)$.

Problem 3.12 $\forall xyP(x,y) \rightarrow \forall xP(x,x)$.

Problem 3.13 $\forall xP(x) \leftrightarrow \forall yP(y)$.

Problem 3.14 $\forall xP(x) \wedge \forall xQ(x) \leftrightarrow \forall x(P(x) \wedge Q(x))$.

Problem 3.15 $\forall xP(x) \vee Q \leftrightarrow \forall x(P(x) \vee Q)$.

Problem 3.16 $(P(a) \vee P(b)) \rightarrow \exists xP(x)$.

Problem 3.17 $(\exists xP(x) \wedge \forall x(P(x) \rightarrow Q(x))) \rightarrow \exists xQ(x)$.

Problem 3.18 $\exists xP(x) \leftrightarrow \exists yP(y)$.

Problem 3.19 $\neg\exists xP(x) \leftrightarrow \forall x\neg P(x)$.

Problem 3.20 $\exists x(P(x) \wedge Q) \leftrightarrow \exists xP(x) \wedge Q$.

Problem 3.21 $\exists x(P(x) \vee Q(x)) \leftrightarrow \exists xP(x) \vee \exists xQ(x)$.

Problem 3.22 $x = y \rightarrow f(x,y) = f(y,x)$.

Problem 3.23 $\forall x\exists y(y = f(x))$.

Problem 3.24 $\exists y(x = y \wedge y = z) \rightarrow x = z$.

Problem 3.25 $(\exists xP(x) \wedge \exists x\neg P(x)) \rightarrow \exists xy(x \neq y)$.

Problem 3.26 $\forall x(x = a \rightarrow P(x)) \leftrightarrow P(a)$.

Problem 3.27 $\exists x(x = a \wedge P(x)) \leftrightarrow P(a)$.