Handout 1*

Syntax and Semantics of Propositional Formulas

Logic is the study of reasoning. Logic can be used, in particular, 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.

In logic, 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 the following, the object language is the formal language of propositional 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.

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

$$\wedge \quad \vee \quad \supset \quad \equiv \quad \neg \quad \bot \quad \top$$

are called *propositional connectives*. Among them, the symbols $\wedge$ *(conjunction)*, $\vee$ *(disjunction)*, $\supset$ *(implication)* and $\equiv$ *(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,

*This note is a part of Vladimir Lifschitz' lecture notes on mathematical logic.

- for any binary connective $\odot$, if F and G are formulas then $(F \odot G)$ is a formula.

For instance,

$$\neg(p \supset q)$$

and

$$(\neg p \supset q) \tag{1.1}$$

are formulas; the string

$$\neg p \supset q \tag{1.2}$$

is not a formula. But very soon (see next page) we are going to introduce a convention according to which (1.2) can be used as an abbreviation for (1.1).

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 $\equiv$ has a lower binding power than the other binary connectives. For instance,

$$p \vee q \equiv p \supset r$$

will be viewed as shorthand for

$$((p \vee q) \equiv (p \supset r)).$$

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

(1.3)

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	$\wedge(x,y)$	$\vee(x,y)$	$\supset(x,y)$	$\equiv(x,y)$
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 (1.3) 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.

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 by Emil Post in 1921. Post is known, along with Alan Turing, as one of the creators of theoretical computer science. He taught at the City College of New York from 1936 until his death in 1954.

Tautologies

A propositional formula F is a *tautology* if every interpretation satisfies F.

Exercise: Check that each of the formulas

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

is a tautology or not.

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 $F \equiv G$ 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:

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

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

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

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

(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

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

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

(g) For each of the formulas

$$p \supset q,\ p \wedge q$$

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

Adequate Sets of Connectives

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

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

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

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 $\{\wedge, \vee, \supset, \equiv, \top\}$ 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* (DNF) 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* (CNF) 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 DNF. Show that $\neg F$ is equivalent to a formula in CNF.

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

Satisfiability and Entailment

A set Γ of formulas is *satisfiable* if there exists an interpretation that satisfies all formulas in Γ, and *unsatisfiable* otherwise. A formula F is *satisfiable* if $\{F\}$ is satisfiable.

Exercise: 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.

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.

Problem 1.12 Prove that $F_1, \ldots, F_n \models G$ iff $(F_1 \wedge \cdots \wedge F_n) \supset G$ is a tautology. (In the first of these expressions, we dropped the braces $\{\ \}$ around $F_1, \ldots, F_n$.)

Problem 1.13 Prove that for any set Γ of formulas and any formula F, $\Gamma \models F$ iff the set $\Gamma \cup \{\neg F\}$ is unsatisfiable.