# *Propositional logic*

*Michael Franke*

Syntax & semantics of propositional logic; truth-tables; tautologies vs. contraditions vs. contingencies; logical equivalence; translations from natural language into propositional logic; semantic meaning vs. pragmatic enrichment; argument schemas & logical validity.

## 1 *The language of propositional logic*

Propositional logic (PropLog) studies how propositions are combined by logical operators, which closely correspond to certain sentential connectives in natural language (such as *and*, *or*, *if*, or *not*). A *proposition* in the sense of PropLog is a minimal unit of thought which can be evaluated as true or false independently of other propositions.[1] For example, the logical structure of the sentence:

$$\underbrace{\text{The earth is round}}_{p}\ \underbrace{\text{and}}_{\wedge}\ \underbrace{\text{the moon is made of cheese.}}_{q}$$

could be analyzed as composed of two propositions, viz., the proposition (denoted here with *proposition letter* $p$) that the earth is round and the proposition (denoted by $q$) that the moon is made of cheese. These two propositions are connected by a logical operator "and," for which we write $\wedge$ in PropLog. The logical structure of the complex sentence above can therefore be written as $p \wedge q$ in PropLog.

[1] The notion of a proposition in this sense is not unproblematic. For example, a case like "*This pixel is red.*" seems like a minimal unit of truth-evaluable information about the color of a particular pixel, but it is not independent of another statement like "*This pixel is blue.*" (Historically, this color-related problem was brought up famously by Frank Ramsey in response to the early logical work of Ludwig Wittgenstein.)

### 1.1 *Proposition letters & sentential connectives*

The language of PropLog is formed by:

(i) a set of *proposition letters* $\mathfrak{P} = \{p, q, r, s, p_1, q_{27}, \dots\}$, and

(ii) a set of *sentential connectives* $\{\neg, \wedge, \vee, \rightarrow, \leftrightarrow\}$.[2]

[2] You will also find terminology like *logical connectives* or *logical operators*. There may be additional connectives used by some logicians or textbooks, and you might find slightly different symbols for the same notions in some places

The sentential connectives have names and are intended to correspond (approximately) to natural language paraphrases:

| name | paraphrase | symbol |
|---|---|---|
| negation | "not" | $\neg$ |
| conjunction | "and" | $\wedge$ |
| disjunction | "or" | $\vee$ |
| implication | "if . . . , then . . . " | $\rightarrow$ |
| equivalence | "if and only of" | $\leftrightarrow$ |

## 1.2 *Formulas*

The language $\mathfrak{L}$ of PropLog is the set of all *formulas* which are recursively defined as follows:[3]

[3] We will make consistent use of Greek letters $\varphi, \psi, \chi, \ldots$ as variables for formulas.

(i) Every proposition letter is a formula.

(ii) If $\varphi$ is a formula, so is $\neg\varphi$.

(iii) If $\varphi$ and $\psi$ are formulas, so are:
a. $(\varphi \wedge \psi)$ b. $(\varphi \vee \psi)$ c. $(\varphi \rightarrow \psi)$ d. $(\varphi \leftrightarrow \psi)$

(iv) Anything that cannot be constructed by (i)–(iii) is not a formula.

Examples for formulas of PropLog are:[4]

[4] We conventionally omit the outermost parentheses of a formula.

$$p \qquad\qquad p \wedge p$$
$$p \rightarrow \neg q \qquad\qquad (p \vee q) \leftrightarrow r$$

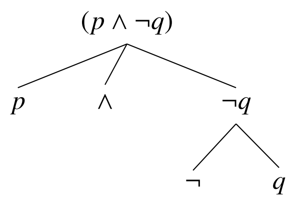
Examples of strings made of proposition letters and logical connectives which are *not* formulas of PropLog are:

$$(p) \qquad\qquad \neg pq$$
$$p\neg \rightarrow \neg q \qquad\qquad p \vee q \leftrightarrow r$$

## 1.3 *Syntactic trees*

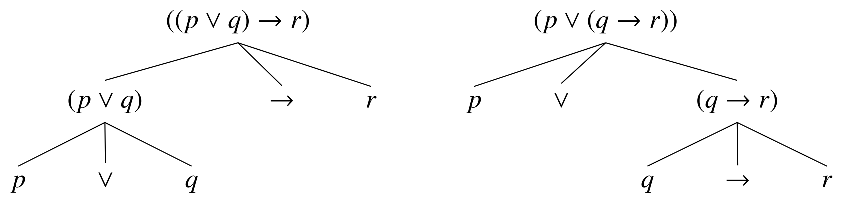
The recursive definition for formulas of PropLog gives an internal structure to each formula. Take the example $p \wedge \neg q$. There is only one way in which this formula could have been generated by a constructive process that follows the recursive definition above. In the last step of that process, the two subformulas $\varphi = p$ and $\psi = \neg q$ have been combined to form an expression of the form $\varphi \wedge \psi$ using rule (iii) part a. The first subformula $\varphi = p$ is constructed by rule (i). The second formula $\psi = \neg q$ can only be constructed by first using rule (i) to introduce $q$ and then using rule (ii) to introduce the negation sign.

A *syntactic tree* is a useful visual illustration of the internal structure of a formula. The syntactic tree of formula $p \wedge \neg q$ is this:



The construction of a complex formula, and therefore its syntactic tree, is always recoverable by following the introduction of the parentheses in step (iii). This is illustrated by the minimal pair in Figure 1.[5]

[5] This is why at most the outer parentheses may be omitted, but never any other pair of parentheses.



(a) Tree for $((p \vee q) \to r)$ (b) Tree for $(p \vee (q \to r))$

Figure 1: Examples of syntactic trees

## 1.4 *Terminology*

A formula of PropLog which consist of a single proposition letter is called *atomic formula.* Any formula of PropLog which is not atomic is called a *complex formula*.

Each complex formula has a *main connective*. The main connective is the last sentential connector introduced during the construction of the formula. A complex formula is also often called by the name of its main connective. For example, the formula $p \wedge \neg q$ has a conjunction as its main operator and would therefore be called a conjunction. The formula $(p \vee q) \to r$ from Figure 1(a) is an implication, while the formula $p \vee (q \to r)$ from Figure 1(b) is a disjunction.

For some connectives, the subformulas that they contain may have special names as well. In the conjunction $\varphi \wedge \psi$, the subformulas $\varphi$ and $\psi$ are called *conjuncts*. In the disjunction $\varphi \vee \psi$, the subformulas $\varphi$ and $\psi$ are called *disjuncts*. In the implication $\varphi \to \psi$, the subformula $\varphi$ is called *antecedent* and the subformula $\psi$ is called *consequent*.

**Exercise 1.** Determine which of the following strings are formulas of propositional logic. For any formula, determine its main operator.

a. $q_{12}$

b. $p, q \wedge r$

c. $(p) \wedge q$

d. $p \to (p \wedge p)$

e. $(p \to)(p \wedge p)$

f. $(p \vee \neg q) \leftrightarrow (r \to (\neg(p \vee \neg p)))$

**Exercise 2.** Draw the syntactic tree for each of the following formulas.

a. $p \leftrightarrow q$

b. $\neg p \wedge p$

c. $p \to \neg(q \wedge r)$

d. $(\neg p \vee \neg q) \wedge (r \to p)$

## 2 The semantics of propositional logic

The language of PropLog defines rules of systematically combining proposition letters with sentential connectives. The *semantics* of PropLog gives a meaning to each formula. Yet, PropLog is not concerned with the meaning of the proposition letters. Whether the proposition $p$ corresponding to "The earth is round." is true or false in this world we live in, is of no concern to the logician. The (propositional) logician cares only about the meaning of the sentential connectives. More specifically, PropLog pays attention to a specific kind of meaning, namely *truth-conditional meaning*. Concretely, PropLog analyzes the meaning of a sentential connective $\odot \in \{\wedge, \vee, \rightarrow, \leftrightarrow\}$ in terms of how the truth or falsity of the complex sentence $\varphi \odot \psi$ depends on the truth or falsity of its components $\varphi$ and $\psi$; and how the meaning of $\neg\varphi$ depends on the truth or falsity of $\varphi$.

### 2.1 Truth tables

One way of defining the semantics of the sentential connectives is by *truth tables*. Truth tables are also handy for systematically calculating the conditions under which complex formulas are true.

Take the complex formula $\neg\varphi$.[6] For all we know, $\varphi$ could be either true or false. There are just these two possibilities.[7] To give a truth-conditional meaning to the operator $\neg$ we need to define whether $\neg\varphi$ is true or false for each case: when $\varphi$ is true, and when $\varphi$ is false. The truth table that defines the meaning of negation is given in Table 1(a).[8] PropLog treats negation like a switch: take a formula $\varphi$ as input, look at its truth value, and swap it for the other (since there are only two truth values).

Other sentential connectives than negation take two formulas as input, so to speak, and may therefore be conceived of as functions that take a pair of truth values as input to return a single truth-value. The semantics of the binary connectives is given in Table 1(b):

[6] We use $\varphi$ here as a variable which could represent any formula, no matter if it is a proposition letter or extremely long and complex.

[7] That PropLog only considers two truth values is also referred to as the *principle of bivalence*. There are other logics, so-called many-valued logics, which allow for more than two truth values, e.g., to represent degrees of truth or uncertainty.

[8] We here write 1 for the truth value "true" and 0 for "false." It is also common to use letters $T$ and $F$, or yet other symbols in truth tables.

| $\varphi$ | $\neg\varphi$ |
|---|---|
| 1 | 0 |
| 0 | 1 |

(a) Semantics of negation

| $\phi$ | $\psi$ | $\phi \wedge \psi$ | $\phi \vee \psi$ | $\phi \rightarrow \psi$ | $\phi \leftrightarrow \psi$ |
|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 1 | 1 |

(b) Semantics of binary connectives

Table 1: Semantics of sentential connectives in propositional logic expressed in terms of truth tables

### 2.2 Working with truth tables

Truth tables are useful for systematically working out the conditions under which complex formulas are true. Consider the complex formula $(\varphi \wedge \psi) \rightarrow$

$\neg\chi$. We do not know whether the subformula $\varphi$, $\psi$ and $\chi$ are atomic or themselves complex. But we want to know under which truth value assignments to $\varphi$, $\psi$ and $\chi$ the formula $(\varphi \wedge \psi) \rightarrow \neg\chi$ is true or false. In order to find out, we can construct a truth table that starts by enumerating all logical possibilities of truth-value assignments for the formulas $\varphi$, $\psi$ and $\chi$ (the first three columns in the table below). We then work towards the "goal formula" $(\varphi \wedge \psi) \rightarrow \neg\chi$ by following the recursive definition of formulas of ProPLog. At each step we apply the definition of the semantics of the relevant sentential connective until we arrive at the "goal formula". Following this method, we construct the truth table in Table 2, which shows that the formula $(\varphi \wedge \psi) \rightarrow \neg\chi$ is always true, except when all three subformulas, $\varphi$, $\psi$ and $\chi$ are all true at the same time.

| $\varphi$ | $\psi$ | $\chi$ | $\varphi \wedge \psi$ | $\neg\chi$ | $(\varphi \wedge \psi) \rightarrow \neg\chi$ |
|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 0 | 0 |
| 1 | 1 | 0 | 1 | 1 | 1 |
| 1 | 0 | 1 | 0 | 0 | 1 |
| 1 | 0 | 0 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 0 | 0 | 1 |
| 0 | 0 | 0 | 0 | 1 | 1 |

Table 2: Truth table for formula $(\varphi \wedge \psi) \rightarrow \neg\chi$

## 2.3 *Contingencies, tautologies & contradictions*

A common application of truth tables is to find out whether a given formula is always true, always false or whether it can sometimes be true and sometimes be false. The truth table in Table 2 shows that the formula $(\varphi \wedge \psi) \rightarrow \neg\chi$ belongs to the last category: there are cases, i.e., assignments of truth values to its components (= rows in the table), where it is true, and there are cases where it is false. A formula which can become true and false for different assignments of truth values to its components is called a *contingency*. A formula which is always true under any constellation of truth values for its components is called a *tautology*. A formula which is false for all ways of assigning truth values to its subformulas is a *contradiction*. We say that a tautology is a formula that is *necessarily true*, or *true by logical necessity*. Similarly, a contradiction can be said to be *necessarily false*, or *false by logical necessity*.[9]

To show that any formula of the form $\varphi \vee \neg\varphi$ is a tautology, we can construct the relevant truth table, shown in Table 3(a), and check whether the column for the "goal formula" contains only ones, no zeros. Similarly, the truth table in Table 3(b) reveals that any formula of the logical form $\varphi \wedge \neg\varphi$ is a contradiction.

[9] Another way of thinking about this is to imagine that you know nothing at all about some foreign universe. You don't know at all which proposition letters are true or not. Given a tautology you can nevertheless be sure that it is true, no matter what the world is like. Of a contradiction you are sure that it is false. And for a contingency you know that you don't know whether it is true or false, before you learn facts about the world.

| $\varphi$ | $\neg\varphi$ | $\varphi \vee \neg\varphi$ |
|---|---|---|
| 1 | 0 | 1 |
| 0 | 1 | 1 |

(a) Truth table for $\varphi \vee \neg\varphi$

| $\varphi$ | $\neg\varphi$ | $\varphi \wedge \neg\varphi$ |
|---|---|---|
| 1 | 0 | 0 |
| 0 | 1 | 0 |

(b) Truth table for $\varphi \wedge \neg\varphi$

Table 3: Truth tables for a tautology and a contradiction

## *2.4 Logical equivalence*

Two formulas $\varphi$ and $\psi$ are logically equivalent if they have exactly the same truth value no matter how we assign meanings to all subformulas of $\varphi$ and $\psi$, as long as we assign any subformula that occurs in both $\varphi$ and $\psi$ the same truth value. We can use truth table to test when or demonstrate that two formulas are logically equivalent. To do this, we just have to make a combined truth table in which we have one column for $\varphi$ and another for $\psi$. $\varphi$ and $\psi$ are logically equivalent exactly when the truth values in the columns that correspond to them are identical in each row. The truth table in Table 4 shows that $\varphi \wedge \neg\psi$ and $\neg(\varphi \to \psi)$ are in fact logically equivalent.

| $\varphi$ | $\psi$ | $\neg\psi$ | $\varphi \wedge \neg\psi$ | $\varphi \to \psi$ | $\neg(\varphi \to \psi)$ |
|---|---|---|---|---|---|
| 1 | 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 1 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 1 | 0 |

Table 4: Truth table showing the equivalence of $\varphi \wedge \neg\psi$ and $\neg(\varphi \to \psi)$

If two formulas $\varphi$ and $\psi$ are *not* logically equivalent, we can also show this with a truth table. In that case, there must be at least one row in which their truth values differ. In practice it may help to mark one such row as the refuting counterexample against the assumption of logical equivalence. This is shown in Table XYZ

| $\varphi$ | $\psi$ | $\varphi \to \psi$ | $\varphi \leftrightarrow \psi$ | counterexample |
|---|---|---|---|---|
| 1 | 1 | 1 | 1 | |
| 1 | 0 | 0 | 0 | |
| 0 | 1 | 1 | 0 | $\Leftarrow$ |
| 0 | 0 | 1 | 1 | |

Table 5: Truth table showing the non-equivalence of $\varphi \to \psi$ and $\varphi \leftrightarrow \psi$

**Exercise 3.** For each of the following formulas, try to intuit whether it is a tautology, a contradiction or a contingency. Write down your best guess. Then use the truth-table method to find out for certain.

a. $\varphi \to \neg\varphi$

b. $\varphi \leftrightarrow (\varphi \wedge \psi)$

c. $(\varphi \wedge \neg\varphi) \to \psi$

d. $(\varphi \wedge \neg\psi) \leftrightarrow (\varphi \to \psi)$

**Exercise 4.** For each of the following pairs of formulas, try to intuit whether they are logically equivalent of not. Write down you guess. Then use truth tables to determine whether they are. Highlight the relevant columns in your truth table. Mark a row as a counterexample if they are not.

a. $\varphi,\ \neg\neg\varphi$

b. $\varphi \leftrightarrow \phi,\ \varphi \wedge \psi$

c. $\neg\varphi \vee \psi,\ \varphi \to \psi$

d. $\varphi \vee (\chi \to \neg\varphi),\ \psi \vee (\chi \to \neg\psi)$

## 3 *Logical structure of meaning*

Linguistic theory distinguishes between *semantic meaning* of a sentence or an expression and additional *pragmatic enrichments* to the literal meaning. The literal meaning is the context-independent logical core meaning. Pragmatic enrichments arise in context by taking into account general world knowledge or the reasons why a speaker used language in the way that they did. Pragmatic enrichments may be part of what we "perceive" to be the most likely interpretation of a sentence; they may be what the speaker meant to say, but not what the sentence means literally.[10]

[10] The most obvious example is an utterance of a sentence like *The weather is so nice today!* when clearly meant ironically.

Our current goal is to learn to excavate the logical structure of natural language sentences. Therefore, we have to learn to "strip off the pragmatic layer," so to speak. Indeed, the logical operators of PropLog bear a close correspondence to natural language expressions *and*, *or*, *if* and *if and only if*. But the correspondence is not a perfect match. This is partly due to the difference between semantic meaning and pragmatic enrichment.[11] Let us look at a few interesting cases.

[11] There are also arguments suggesting that the logical operators of PropLog are *not* good enough for analyzing the semantic meaning of their corresponding natural language expressions. Most importantly, this concerns the semantic meaning of conditional sentences, i.e., natural language sentences with *if dots, then* . . . .

### 3.1 *Conjunction & order-sensitivity*

Logical conjunction is order-insensitive: $\varphi \wedge \psi$ and $\psi \wedge \varphi$ are logically equivalent. But natural language conjunctions are not necessarily. Saying that

*They got married and had children.*

might be understood differently from saying that

*They had children and got married.*

Still, we would analyze both of these sentences as having a logical-conjunctive meaning, additional inferences about the temporal order of events notwithstanding.

### 3.2 *Disjunction: inclusive vs. exclusive*

Logical disjunction is *inclusive*: $\varphi \vee \psi$ is true when $\varphi$ and $\psi$ are both true. Yet natural language disjunctions *can* be understood as exclusive disjunction according to which exactly one, but not both disjuncts are true. For example, if I say:

*She owns a Porsche or a Ferrari.*

you might take that to mean that she doesn't own both. But did I *literally* say that? — The logical meaning of that sentence is more plausibly just analyzed as inclusive disjunction. This is because it is not contradictory to say:

*She owns a Porsche or a Ferrari and possibly both.*

This contrasts with the weirdness of saying:

*She owns* either *a Porsche or a Ferrari and possibly both.*

suggesting that natural language *either . . . or . . .* *does* have an exclusive disjunctive meaning, but simple *or* does not.

## 3.3 *Conditional perfection*

PropLog analyzes implication as so-called *material implication*, but it is controversial whether natural language conditional sentences might not have a much richer meaning, e.g., requiring a discernible causal-inferential relation between antecedent and consequent. Moreover, similar to the case of inclusive vs. exclusive disjunctions, many natural language conditionals receive a so-called *conditional perfection reading*, i.e., they are read like an "if and only if" statement which would correspond to logical operator $\leftrightarrow$, even though their logical meaning might just be that of a simple implication $\rightarrow$. For example, the conditional statement:

*If Bubu comes, Kiki comes.*

does suggest to a certain degree that Kiki *only* comes when Bubu does. But that is not the logical core meaning of this sentence, but rather a pragmatic enrichment on top of the logical meaning of the sentence. To see this, compare the difference in weirdness between saying

*If Bubu comes, Kiki comes, and maybe Kiki comes no matter what.*

which seems fine, especially when compared to the much weirder:

*Kiki comes only if Bubu comes, and maybe Kiki comes not matter what.*

## 3.4 *Excavating the logical structure of natural language sentences*

Despite superficial discrepancies, it is possible and highly informative to try to lay bare the logical structure of natural language sentences.The logical relation between propositions contained in a natural language sentence may sometimes be opaque. Take the following example sentence:

*If drawing a red card means that I lost, I lost.*

This sentences does have a propositional-logical structure, even though it is not at all apparent from the way it is realized in natural language. There are two atomic (truth-evaluable) propositions involved here:

| | |
|---|---|
| $p$: I drew a red card. | $q$: I lost. |

With this *translation key*, we can then determine the logical structure of the sentence above as:

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

Here are a few other examples.

(i) John is hungry.
*Key:* $p$: John is hungry. *Logical form:* $p$

(ii) Mary likes John.
*Key:* $p$: Mary likes John. *Logical form:* $p$

(iii) John is hungry and Mary likes John.
*Key:* $p$: John is hungry. $q$: Mary likes John.
*Logical form:* $p \wedge q$

(iv) Berlin is east of Hamburg and Bremen.
*Key:* $p$: Berlin is east of Hamburg. $q$: Berlin is east of Bremen.
*Logical form:* $p \wedge q$

(v) Call me 'sweetie' once more and, no kidding, I'll step on your toe.
*Key:* $p$: I'm not kidding. $q$: I'll step on your toe.
$r$: You call me 'sweetie' once more.
*Logical form:* $p \wedge (r \rightarrow q)$

(vi) I will only play salsa, if you give me earplugs.[12]
*Key:* $p$: I will play salsa. $q$: You give me earplugs.
*Logical form:* $p \rightarrow q$

[12]There is a likely inclination to analyze this as $p \leftrightarrow q$, but it is not clear whether this is part of the semantic/logical content here. Suppose you know that the sentence is true. Now you learn that earplugs have been handed over. Does that *logically* entail that salsa was played? No, because the sentence only says that earplugs are a *necessary* but not necessarily *sufficient* condition for salsa.

**Exercise 5.** For each of the following sentences give the translation key and the (propositional) logical form.

(a.) If John is hungry, then Mary likes John.

(b.) If John is hungry and Bill runs, then Mary doesn't likes John.

(c.) John is hungry or it's false that Bill runs.

(d.) Mary likes John or Mary doesn't like John.

(e.) Mary likes John but John is hungry.

(f.) Mary likes John provided Bill runs.

## *4 Valuation functions, possible worlds & logical entailment*

TBD