

# Logic

Logic deals with the formalization of natural language and reasoning methods.

A variety of logical systems have been developed, including

- propositional logic,
- predicate logic,
- temporal logics, and
- modal logics.

Typical applications of logic in computing include

- artificial intelligence,
- reasoning about knowledge,
- logic programming,
- and automated verification.

We will begin with an introduction to the *logic of compound statements*, which is a basis for many other logical systems and is also called *propositional logic*.

## From the Files of Inspector Craig

The following facts are known about a robbery:

1. If A is guilty and B is innocent, then C is guilty.
2. C never works alone.
3. A never works with C.
4. No one other than A, B, or C was involved, and at least one of them is guilty.

Can one infer from these facts who is guilty and who is innocent?

# The Case of McGregor's Shop

Mr. McGregor phoned Scotland Yard that his shop had been robbed. Three suspects A, B, C were rounded up for questioning and the following facts were established:

1. Each of A, B, C had been in the shop on the day of the robbery, and no one else had been in the shop that day.
2. If A was guilty, then he had exactly one accomplice.
3. If B is innocent, so is C.
4. If exactly two are guilty, then A is one of them.
5. If C is innocent, so is B.

Whom did Inspector Craig indict?

These, and other problems, are discussed in *What Is the Name of This Book?* by Raymond Smullyan.

# Fundamental Notions in Logic

The above examples can be formalized in propositional logic, a system based on well-known logical connectives such as negation, conjunction, and disjunction.

Key questions in the study of logical systems include the following:

*Validity* – When is a given logical formula true?

*Logical consequence* – Does a given statement necessarily follow from given assumptions?

*Provability* – Can a desired conclusion be syntactically derived from given axioms?

We will consider these questions in the context of propositional logic, but the relationship between *truth* and *proof* is a central issue in the study of logical systems in general.

# Propositional Logic

Propositional logic is a formal system in which the basic units are *statements* (or *propositions*).

The basic assumption is that

*each proposition represents a sentence that is either true or false (but not both).*

Simple propositions are denoted by (*propositional*) *variables* or by constants representing truth or falsity. Statements can be combined via *logical connectives* into more complex, *compound* propositions.

The connectives used to form more complex propositions include negation ( $\sim$ , read “not”), conjunction ( $\wedge$ , read “and”), disjunction ( $\vee$ , read “or”), and implication ( $\rightarrow$ , read “if-then”).

# Syntax

The following rules specify the syntax of propositional formulas:

$$\begin{aligned} \langle \text{proposition} \rangle & ::= \top \mid \perp \mid \langle \text{variable} \rangle \\ & \mid (\sim \langle \text{proposition} \rangle) \\ & \mid (\langle \text{proposition} \rangle \wedge \langle \text{proposition} \rangle) \\ & \mid (\langle \text{proposition} \rangle \vee \langle \text{proposition} \rangle) \\ & \mid (\langle \text{proposition} \rangle \rightarrow \langle \text{proposition} \rangle) \end{aligned}$$

$$\langle \text{variable} \rangle ::= p \mid q \mid r \mid \dots$$

We use the letters  $\alpha$  and  $\beta$  to denote propositional formulas.

Other common connectives are exclusive disjunction ( $\oplus$ , read “either-or”) and biconditional ( $\leftrightarrow$ , read “if and only if”).

Parentheses are often omitted to increase readability, but one has to be careful to avoid ambiguous expressions.

# Semantics

The constants  $\top$  and  $\perp$  are also called *truth values* and represent truth and falsity, respectively.

The semantics of propositional logic formulas rests on so-called *truth functions*. These functions are usually defined by *truth tables* that specify the necessary identities for each connective.

For negation we have the following identities:

$$\begin{aligned}\sim\top &= \perp \\ \sim\perp &= \top\end{aligned}$$

In other words, the negation of a true statement is false, and vice versa.

Conjunction and disjunction are defined by:

$$\begin{array}{ll} \top \wedge \top &= \top & \top \vee \top &= \top \\ \top \wedge \perp &= \perp & \top \vee \perp &= \top \\ \perp \wedge \top &= \perp & \perp \vee \top &= \top \\ \perp \wedge \perp &= \perp & \perp \vee \perp &= \perp \end{array}$$

That is, a conjunction is true if, and only if, both arguments are true; while a disjunction is true if, and only if, at least one argument is true.

*Conditional* statements, of the form  $\alpha \rightarrow \beta$ , will be discussed in detail later. For now let  $\alpha \rightarrow \beta$  be an abbreviation of  $\sim\alpha \vee \beta$ .

# Inspector Craig's Case

The known facts can be represented by the formulas

1.  $p \wedge \sim q \rightarrow r$

2.  $r \rightarrow p \vee q$

3.  $\sim(p \wedge r)$

4.  $p \vee q \vee r$

where  $p$  represents the statement "A is guilty,"  $q$  the statement "B is guilty," and  $r$  the statement "C is guilty."

Let  $\alpha$  be the conjunction of the above four formulas. We get the following truth table for  $\alpha$ :

$p$	$q$	$r$	$\alpha$
⊥	⊥	⊥	⊥
⊥	⊥	⊤	⊥
⊥	⊤	⊥	⊤
⊥	⊤	⊤	⊤
⊤	⊥	⊥	⊥
⊤	⊥	⊤	⊥
⊤	⊤	⊥	⊤
⊤	⊤	⊤	⊥

# Truth Valuations

The construction of a truth table requires two steps for each entry: (i) application of a so-called “truth valuation” and (ii) evaluation of the resulting variable-free formula using truth functions.

Formally, a (truth) *valuation* is a mapping from propositional variables to truth values.

Various notations are used to denote such mappings, e.g.,

$$[p \mapsto \top, q \mapsto \top, r \mapsto \perp]$$

or

$$[\top/p, \top/q, \perp/r].$$

If  $\alpha = \alpha(p_1, \dots, p_n)$  is a formula containing variables  $p_1, \dots, p_n$ , and  $\sigma$  is a truth valuation, then by  $\alpha\sigma$  we denote the result of replacing in  $\alpha$  each occurrence of a variable  $p_i$  by its truth value, as specified by  $\sigma$ .

# Tautologies and Contradictions

A propositional formula  $\alpha$  is said to be *satisfiable* if it evaluates to  $\top$  for some truth valuation.

A formula  $\alpha$  is called a *tautology* if it evaluates to  $\top$  for all truth valuations.

Finally,  $\alpha$  is called a *contradiction* (or *unsatisfiable*) if it always evaluates to false.

A formula that is neither a tautology nor a contradiction is also called a *contingency*.

For example,  $p \vee \sim p$  is a tautology, whereas  $p \wedge \sim p$  is a contradiction.

**Theorem** [Tautology and contradiction]

A propositional formula  $\alpha$  is a tautology if and only if its negation  $\sim\alpha$  is a contradiction.

# Logical Equivalence

Two propositional formulas  $\alpha$  and  $\beta$  are said to be *equivalent*, written  $\alpha \equiv \beta$ , if and only if  $\alpha\sigma$  and  $\beta\sigma$  evaluate to the same truth value for each truth valuation  $\sigma$ .

In general, one may check whether two formulas are equivalent by constructing and comparing their respective truth tables. (*Caveat*: It is possible for two formulas with different variables to be equivalent.)

## Examples:

$$\begin{aligned}\sim p \rightarrow q &\equiv p \vee q \\ p \wedge \sim p &\equiv \sim(p \rightarrow p) \\ p \rightarrow p &\equiv \top\end{aligned}$$

Another way of verifying an equivalence is to run a tautology check for a suitable formula.

## Theorem [Tautology and equivalence]

Two propositional formulas  $\alpha$  and  $\beta$  are logically equivalent if and only if the formula

$$(\alpha \rightarrow \beta) \wedge (\beta \rightarrow \alpha)$$

is a tautology.

# Basic Equivalences

$$\begin{aligned} p \wedge q &\equiv q \wedge p \\ p \vee q &\equiv q \vee p \\ (p \wedge q) \wedge r &\equiv p \wedge (q \wedge r) \\ (p \vee q) \vee r &\equiv p \vee (q \vee r) \\ p \wedge (q \vee r) &\equiv (p \wedge q) \vee (p \wedge r) \\ p \vee (q \wedge r) &\equiv (p \vee q) \wedge (p \vee r) \\ p \wedge \top &\equiv p \\ p \vee \perp &\equiv p \\ p \vee \sim p &\equiv \top \\ p \wedge \sim p &\equiv \perp \\ \sim \sim p &\equiv p \\ p \wedge p &\equiv p \\ p \vee p &\equiv p \\ p \vee \top &\equiv \top \\ p \wedge \perp &\equiv \perp \\ \sim(p \wedge q) &\equiv \sim p \vee \sim q \\ \sim(p \vee q) &\equiv \sim p \wedge \sim q \\ p \vee (p \wedge q) &\equiv p \\ p \wedge (p \vee q) &\equiv p \end{aligned}$$

# Conditional Statements

Statements of the form

*if  $\alpha$  then  $\beta$*

are said to be *conditional*.

Conditional statements are represented via another binary connective,

$\alpha \rightarrow \beta$ ,

the semantics of which is defined by the following identities:

$$\begin{array}{l} \top \rightarrow \top = \top \\ \top \rightarrow \perp = \perp \\ \perp \rightarrow \top = \top \\ \perp \rightarrow \perp = \top \end{array}$$

Thus, a conditional is false in only one case, when the antecedent is true but the consequent false. In all other cases the conditional is true.

The formula  $\alpha$  is called the *hypothesis* (or *antecedent*) of the conditional and  $\beta$  the *conclusion* (or *consequent*).

# Equivalences for Conditionals

A conditional formula

$$\alpha \rightarrow \beta$$

is logically equivalent to a disjunction

$$\sim\alpha \vee \beta.$$

In other words, conditional statements can also be expressed via negation and disjunction.

The formula  $\alpha \rightarrow \beta$  is also equivalent to  $\sim(\alpha \wedge \sim\beta)$ , and consequently its negation,  $\sim(\alpha \rightarrow \beta)$ , is equivalent to a conjunction  $\alpha \wedge \sim\beta$ .

The formula  $\sim\beta \rightarrow \sim\alpha$  is called the *contrapositive* of  $\alpha \rightarrow \beta$ .

## Theorem

A conditional formula is logically equivalent to its contrapositive.

# Converse and Inverse

If we switch the arguments of  $\alpha \rightarrow \beta$ , we obtain what is known as its *converse*,  $\beta \rightarrow \alpha$ .

For example, consider the statement

*If you have perfect scores on all exams and assignments, then you get an A for the course.*

Its converse is

*If you get an A for the course, then you have perfect scores on all exams and assignments.*

Evidently, the converse is *not* logically equivalent to the original statement.

The *inverse* of  $\alpha \rightarrow \beta$  is defined to be  $\sim\alpha \rightarrow \sim\beta$ .

Thus, the inverse of the above statement is

*If you do not have perfect scores on all exams and assignments, then you do not get an A for the course.*

## **Theorem.**

The inverse and converse of a conditional statement are equivalent to each other.

## Necessary and Sufficient Conditions

If a conditional  $\alpha \rightarrow \beta$  is true then  $\alpha$  is said to be a *sufficient condition* for  $\beta$ .

For example,

*You have perfect scores on all exams and assignments.*

is a sufficient condition for

*You get an A for the course.*

We say that  $\alpha$  is a *necessary condition* for  $\beta$  if  $\beta \rightarrow \alpha$  is true.

The example shows that sufficient conditions need not be necessary, but one necessary condition for

*You get an A for the course.*

is

*You pass the final exam.*

# Biconditionals

Another common propositional connective is the *biconditional*, which we denote by the symbol  $\leftrightarrow$  and defined by the identities:

$$\begin{aligned} \top \leftrightarrow \top &= \top \\ \top \leftrightarrow \perp &= \perp \\ \perp \leftrightarrow \top &= \perp \\ \perp \leftrightarrow \perp &= \top \end{aligned}$$

That is,  $p \leftrightarrow q$  is true if, and only if,  $p$  and  $q$  are either both true or both false.

We have the following equivalence:

$$(p \leftrightarrow q) \equiv (q \rightarrow p) \wedge (p \rightarrow q).$$

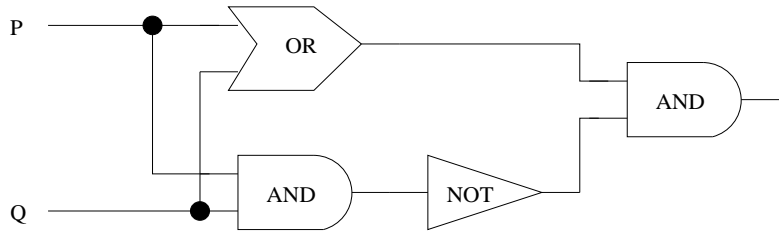
That is, a biconditional semantically corresponds to a conjunction of two conditionals, which are called its *if-part* and *only-if-part*, respectively.

# Digital Logic Circuits

The principles of propositional logic can be applied to digital logic circuits, as the input-output behavior of the basic components, which are called *logic gates*, can be described by Boolean (or truth) functions.

If a circuit is constructed so that no output of a gate can eventually feed back into that gate, then its behavior can also be represented by a Boolean function.

For example, the circuit



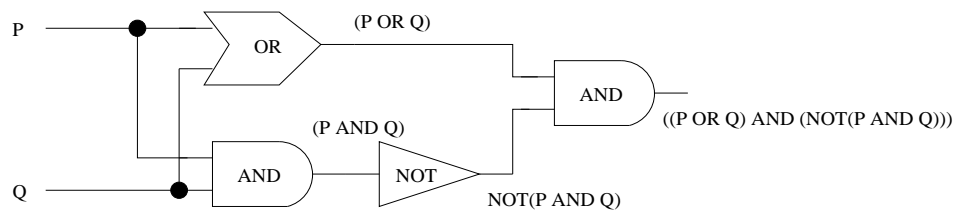
computes a function corresponding to exclusive disjunction:

$p$	$q$	$f(p, q)$
T	T	$\perp$
T	$\perp$	T
$\perp$	T	T
$\perp$	$\perp$	$\perp$

# Formulas and Circuits

The logic formula representing the *output function* of a digital logic circuit can be constructed by tracing from the input signals and appropriately enclosing subexpressions within parentheses.

For example, for the above circuit we obtain the formula  $((p \vee q) \wedge \sim(p \wedge q))$ .



This process can be described *recursively*. To obtain the formula for a gate  $G$ ,

1. first construct the formula for the gate leading into the first input to  $G$  and, if applicable, the formula for the second input to  $G$  (each enclosed within parentheses) and
2. then combine the formula(s) with the logical operator corresponding to the type of gate  $G$  ( $\wedge$  or  $\vee$  or  $\sim$ ).

If the input is a simple signal, instead of a gate, represent it by a variable.

This recursive procedure works correctly if the circuit contains no feedback connections, but would result in an infinite loop for circuits with a feedback cycle.

## Other Logical Connectives

The logical system we have discussed is based on a few selected connectives, the semantics of which are defined by truth functions (or truth tables). Conversely, each truth function (or truth table) can be interpreted as defining a logical connective.

For example, the identities

$$\begin{array}{l} \top | \top = \perp \\ \top | \perp = \top \\ \perp | \top = \top \\ \perp | \perp = \top \end{array}$$

define another connective that is known as *Sheffer stroke*. It represents the negation of conjunction (and hence the input-output behavior of a *NAND*-gate):

$$p|q \equiv \sim(p \wedge q).$$

The Sheffer stroke does not increase the expressiveness of our logical system, but are there other connectives that can not be expressed (in equivalent form) with our selected connectives?

# Adequacy

It turns out that the three connectives  $\sim$ ,  $\wedge$  and  $\vee$  form what is called an *adequate* (or *complete*) set, in that every other logical connective defined by truth functions can be expressed in terms of these three.

For example, consider the truth table

$p$	$q$	$r$	$\alpha$
$\perp$	$\perp$	$\perp$	$\perp$
$\perp$	$\perp$	$\top$	$\perp$
$\perp$	$\top$	$\perp$	$\top$
$\perp$	$\top$	$\top$	$\top$
$\top$	$\perp$	$\perp$	$\perp$
$\top$	$\perp$	$\top$	$\perp$
$\top$	$\top$	$\perp$	$\top$
$\top$	$\top$	$\top$	$\perp$

One formula  $\alpha$ , containing only the connectives  $\sim$ ,  $\wedge$ , and  $\vee$  and the variables  $p$ ,  $q$ , and  $r$ , that has this truth table is

$$(\sim p \wedge q \wedge \sim r) \vee (\sim p \wedge q \wedge r) \vee (p \wedge q \wedge \sim r).$$

Thus, the (ternary) connective specified by the identities in the above truth table can also be defined via the formula  $\alpha$ .

## Adequate Sets

Since  $\{\sim, \wedge, \vee\}$  is an adequate set of connectives, and conjunction can be expressed by negation and disjunction, and disjunction by negation and conjunction, we may conclude that the sets  $\{\sim, \vee\}$  and  $\{\sim, \wedge\}$  are also adequate.

The following two equivalences indicate that the set consisting of just the Sheffer stroke is also adequate:

$$\begin{aligned}\sim p &\equiv p|p \\ p \wedge q &\equiv (p|q)|(p|q)\end{aligned}$$

On the other hand, the set  $\{\wedge, \vee\}$  is not adequate:

A formula  $\alpha$  containing only  $\wedge$ ,  $\vee$ , and variables evaluates to  $\top$  if all variables are replaced by  $\top$ . Thus no such formula is equivalent to  $\sim p$ , and hence negation can not be expressed by conjunction and disjunction.