Formal Logic

en.wikibooks.org

May 21, 2017

Contents

Preliminaries

1 Sets

Presentations of logic vary in how much **set theory** they use. Some are heavily laden with set theory. Though most are not, it is nearly impossible to avoid it completely. It will not be a very important focal point for this book, but we will use a little set theory vocabulary here and there. This section introduces the vocabulary and notation used.

1.1 Sets and elements

Mathematicians use 'set' as an undefined primitive term. Some authors resort to quasi-synonyms such as 'collection'.

A *set* has *elements*. 'Element' is also undefined in set theory. We say that an element *is a member of* a set, also an undefined expression. The following are all used synonymously:

x is a member of y

x is contained in y

x is included in y

y contains x

y includes x

1.1.1 Notation

A set can be specified by enclosing its members within curly braces.

$$\{1,2,3\}$$

is the set containing 1, 2, and 3 as members. The curly brace notation can be extended to specify a set using a rule for membership.

$\{x : x = 1 \text{ or } x = 2 \text{ or } x = 3\}$ (The set of all x such that $x = 1$ or $x = 2$ or $x = 3$)

is again the set containing 1, 2, and 3 as members.

$\{x : x \text{ is a positive integer}\}$, and

$$\{1,2,3,\ldots\}$$

both specify the set containing 1, 2, 3, and onwards.

A modified epsilon is used to denote set membership. Thus

$$x \in y$$

indicates that "x is a member of y". We can also say that "x is not a member of y" in this way:

$$x \notin y$$

1.1.2 Characteristics of sets

A set is uniquely identified by its members. The expressions

$$\{x : x \text{ is an even prime}\}$$

$$\{x : x \text{ is a positive square root of } 4\}$$

$$\{2\}$$

all specify the same set even though the concept of an even prime is different from the concept of a positive square root. Repetition of members is inconsequential in specifying a set. The expressions

$$\{1,\ 2,\ 3\}$$

$$\{1,\ 1,\ 1,\ 1,\ 2,\ 3\}$$

$$\{x : \text{x is an even prime or } x \text{ is a positive square root of 4 or } x = 1 \text{ or } x = 2 \text{ or } x = 3\}$$

all specify the same set.

Sets are unordered. The expressions

$$\{1,\ 2,\ 3\}$$

$$\{3,\ 2,\ 1\}$$

$$\{2,\ 1,\ 3\}$$

all specify the same set.

Sets can have other sets as members. There is, for example, the set

$$\{\{1,\ 2\},\{2,\ 3\},\ \{1,\ \text{George Washington}\}\}$$

1.1.3 Some special sets

As stated above, sets are defined by their members. Some sets, however, are given names to ease referencing them.

The set with no members is the *empty set*. The expressions

$$\{\}$$

$$\varnothing$$

$$\{x : x \neq x\}$$

all specify the empty set. Empty sets can also express oxymora ("four-sided triangles" or "birds with radial symmetry") and factual non-existence ("the King of Czechoslovakia in 1994").

A set with exactly one member is called a *singleton*. A set with exactly two members is called a *pair*. Thus $\{1\}$ is a singleton and $\{1,\ 2\}$ is a pair.

ω is the set of natural numbers, $\{0,\ 1,\ 2,\ ...\}$.

1.2 Subsets, power sets, set operations

1.2.1 Subsets

A set s is a *subset* of set a if every member of s is a member of a. We use the horseshoe notation to indicate subsets. The expression

$$\{1,\ 2\} \subseteq \{1,\ 2,\ 3\}$$

says that $\{1,\ 2\}$ is a subset of $\{1,\ 2,\ 3\}$. The empty set is a subset of every set. Every set is a subset of itself. A *proper subset* of a is a subset of a that is not identical to a. The expression

$$\{1,\ 2\} \subset \{1,\ 2,\ 3\}$$

says that $\{1, 2\}$ is a proper subset of $\{1, 2, 3\}$.

1.2.2 Power sets

A *power set* of a set is the set of all its subsets. A script 'P' is used for the power set.

$$\mathcal{P}\{1,\ 2,\ 3\} = \{\varnothing,\ \{1\},\ \{2\},\ \{3\},\ \{1,\ 2\},\ \{1,\ 3\},\ \{2,\ 3\},\ \{1,\ 2,\ 3\}\}$$

1.2.3 Union

The *union* of two sets a and b, written $a \cup b$, is the set that contains all the members of a and all the members of b (and nothing else). That is,

$$a \cup b = \{x : x \in a \text{ or } x \in b\}$$

As an example,

$$\{1,\ 2,\ 3\}\ \cup\ \{2,\ 3,\ 4\} = \{1,\ 2,\ 3,\ 4\}$$

1.2.4 Intersection

The *intersection* of two sets a and b, written $a \cap b$, is the set that contains everything that is a member of both a and b (and nothing else). That is,

$$a \cap b = \{x : x \in a \text{ and } x \in b\}$$

As an example,

$$\{1,\ 2,\ 3\}\ \cap\ \{2,\ 3,\ 4\} = \{2,\ 3\}$$

1.2.5 Relative complement

The *relative complement* of a in b, written $b \setminus a$ (or $b - a$) is the set containing all the members of b that are not members of a. That is,

$$b \setminus a = \{x : x \in b \text{ and } x \notin a\}$$

As an example,

$$\{2,\ 3,\ 4\} \setminus \{1,\ 3\} = \{2,\ 4\}$$

1.3 Ordered sets, relations, and functions

The intuitive notions of *ordered set*, *relation*, and *function* will be used from time to time. For our purposes, the intuitive mathematical notion is the most important. However, these intuitive notions can be defined in terms of sets.

1.3.1 Ordered sets

First, we look at ordered sets. We said that sets are unordered:

$$\{a,\ b\} = \{b,\ a\}$$

But we can define ordered sets, starting with ordered pairs. The angle bracket notation is used for this:

$$\langle a,\ b\rangle\ \neq\ \langle b,\ a\rangle$$

Indeed,

$$\langle x,\ y\rangle\ =\ \langle u,\ v\rangle \text{ if and only if } x = u \text{ and } y = v$$

Any set theoretic definition giving $\langle a,\ b\rangle$ this last property will work. The standard definition of the *ordered pair* $\langle a,\ b\rangle$ runs:

$$\langle a,\ b\rangle\ =\ \{\{a\},\ \{a,\ b\}\}$$

This means that we can use the latter notation when doing operations on an ordered pair.

There are also bigger ordered sets. The *ordered triple* $\langle a,\ b,\ c\rangle$ is the ordered pair $\langle\langle a,\ b\rangle,\ c\rangle$. The *ordered quadruple* $\langle a,\ b,\ c,\ d\rangle$ is the ordered pair $\langle\langle a,\ b,\ c\rangle,\ d\rangle$. This, in turn, is the ordered triple $\langle\langle\langle a,\ b\rangle,\ c\rangle,\ d\rangle$. In general, an *ordered n-tuple* $\langle a_1,\ a_2,\ ...,\ a_{\mathrm{n}}\rangle$ where n greater than 1 is the ordered pair $\langle\langle a_1,\ a_2,\ ...,\ a_{n\text{-}1}\rangle,\ a_{\mathrm{n}}\rangle$.

It can be useful to define an *ordered 1-tuple* as well: $\langle a\rangle\ =\ a$.

These definitions are somewhat arbitrary, but it is nonetheless convenient for an n-tuple, $n\ \rangle\ 2$, to be an n-1 tuple and indeed an ordered pair. The important property that makes them serve as ordered sets is:

$$\langle x_1,\ x_2,\ ...,\ x_n\rangle\ =\ \langle y_1,\ y_2,\ ...,\ y_n\rangle \text{ if and only if } x_1 = y_1,\ x_2 = y_2,\ ...,\ x_n = y_n$$

1.3.2 Relations

We now turn to relations. Intuitively, the following are relations:

$x < y$

x is a square root of y

x is a brother of y

x is between y and z

The first three are binary or 2-place relations; the fourth is a ternary or 3-place relation. In general, we talk about n-ary relations or n-place relations.

First consider binary relations. A *binary relation* is a set of ordered pairs. The *less than* relation would have among its members $\langle 1,\ 2\rangle$, $\langle 1,\ 3\rangle$, $\langle 16,\ 127\rangle$, etc. Indeed, the *less than* relation defined on the natural numbers ω is:

$$\{\langle x,\ y\rangle : x \in \omega,\ y \in \omega,\ \text{and } x < y\}$$

Intuitively, $\langle x,\ y\rangle$ is a member of the *less than* relation if $x < y$. In set theory, we do not worry about whether a relation matches an intuitive concept such as *less than*. Rather, *any* set of ordered pairs is a binary relation.

We can also define a 3-place relation as a set of 3-tuples, a 4-place relation as a set of 4-tuples, etc. We only define n-place relations for $n \geq 2$. An n-place relation is said to have an *arity* of n. The following example is a 3-place relation.

$$\{\langle 1,\ 2,\ 3\rangle,\ \langle 8,\ 2,\ 1\rangle,\ \langle 653,\ 0,\ 927\rangle\}$$

Because all n-tuples where $n > 1$ are also ordered pairs, all n-place relations are also binary relations.

1.3.3 Functions

Finally, we turn to functions. Intuitively, a function is an assignment of values to arguments such that each argument is assigned at most one value. Thus the *+ 2* function assigns a numerical argument x the value $x + 2$. Calling this function f, we say $f(x) = x + 2$. The following define specific functions.

$$f_1(x) = x \times x$$

$$f_2(x) = \text{ the smallest prime number larger than } x$$

$$f_3(x) = 6/x$$

$$f_4(x) = \text{ the father of x}$$

Note that f_3 is undefined when $x = 0$. According to biblical tradition, f_4 is undefined when $x =$ Adam or $x =$ Eve. The following do not define functions.

$$f_5(x) = \pm\sqrt{x}$$

$$f_6(x) = \text{ a son of x}$$

Neither of these assigns unique values to arguments. For every positive x, there are two square roots, one positive and one negative, so f_5 is not a function. For many x, x will have multiple sons, so f_6 is not a function. If f_6 is assigned the value *the son of x* then a unique value is implied by the rules of language, therefore f_6 will be a function.

A *function* f is a binary relation where, if $\langle x,\ y\rangle$ and $\langle x,\ z\rangle$ are both members of f, then $y = z$.

We can define many place functions. Intuitively, the following are definitions of specific many place functions.

$$f_7(x,\ y) = x + y$$

$$f_8(x,\ y,\ z) = (x + y) \times z$$

Thus $\langle 4,\ 7,\ 11\rangle$ is a member of the 2-place function f_7. $\langle 3,\ 4,\ 5,\ 35\rangle$ is a member of the 3 place function f_8

The fact that all n-tuples, $n \geq 2$, are ordered pairs (and hence that all n-ary relations are binary relations) becomes convenient here. For $n \geq 1$, an n-place function is an n+1 place relation that is a 1-place function. Thus, for a 2-place function f,

$$\langle x,\ y,\ z_1\rangle\ \in f \text{ and } \langle x,\ y,\ z_2\rangle\ \in f \text{ if and only if } z_1 = z_2$$

Sentential Logic

Sentential Logic

1. REDIRECT Formal Logic/Sentential Logic/Goals[1]

1 https://en.wikibooks.org/wiki/Formal%20Logic%2FSentential%20Logic%2FGoals

2 The Sentential Language

This page informally describes our sentential language which we name $\mathcal{L}_{\mathcal{S}}$. A more formal description will be given in Formal Syntax[1] and Formal Semantics[2]

2.1 Language components

2.1.1 Sentence letters

The *sentence letters* are single letters such as

$$\text{P},\ \text{Q},\ \text{R},\ \text{etc.}$$

Some texts restrict this to lower case letters, and others restrict them to capital letters. We will use capital letters.

Intuitively, we can think of sentence letters as translating English sentences that are either true or false. Thus, P can translate 'The Earth is a planet' (which is true) or 'The moon is made of green cheese' (which is false). But P can not translate 'Great ideas sleep furiously' because it is neither true nor false. Translations between English and $\mathcal{L}_{\mathcal{S}}$ work best if we restrict ourselves to timelessly true or false present tense[3] sentences in the indicative mood[4]. You will see from the translation section below[5] that we do not always follow that advice. The truth or falsity of those sentences is not timeless.

2.1.2 Sentential connectives

Sentential connectives are special symbols in Sentential Logic that represent truth functional relations. They are used to build larger sentences from smaller sentences. The truth or falsity of the larger sentence can then be computed from the truth or falsity of the smaller ones.

Conjunction: $\land$

- Translates to English as 'and'.
- $\text{P} \land \text{Q}$ is called a *conjunction* and P and Q are its *conjuncts*.

1 Chapter 2.4 on page 15
2 Chapter 4.3 on page 24
3 https://en.wikipedia.org/wiki/Present%20tense
4 https://en.wikipedia.org/wiki/Indicative%20mood%23Indicative_mood
5 Chapter 2.3 on page 14

- $\mathrm{P} \land \mathrm{Q}$ is true if both P and Q are true—and is false otherwise.
- Some authors use an **&** (ampersand), • (heavy dot) or juxtaposition. In the last case, an author would write

$$\mathrm{PQ}$$

instead of our

$$\mathrm{P} \land \mathrm{Q} .$$

Disjunction: $\lor$

- Translates to English as 'or'.
- $\mathrm{P} \lor \mathrm{Q}$ is called a *disjunction* and P and Q are its *disjuncts.*
- $\mathrm{P} \lor \mathrm{Q}$ is true if at least one of P and Q are true—is false otherwise.
- Some authors may use a vertical stroke: |. However, this comes from computer languages rather than logicians' usage. Logicians normally reserve the vertical stroke for nand (alternative denial). When used as nand, it is called the Sheffer stroke[6].

Negation: $\neg$

- Translates to English as 'it is not the case that' but is normally read 'not'.
- $\neg\mathrm{P}$ is called a *negation.*
- $\neg\mathrm{P}$ is true if P is false—and is false otherwise.
- Some authors use ~ (tilde) or $-$. Some authors use an overline, for example writing

$$\bar{\mathrm{P}} \quad \text{and} \quad (\overline{(\mathrm{P} \land \mathrm{Q})} \lor \mathrm{R})$$

instead of

$$\neg\mathrm{P} \quad \text{and} \quad (\neg(\mathrm{P} \land \mathrm{Q}) \lor \mathrm{R}) .$$

Conditional: $\rightarrow$

- Translates to English as 'if...then' but is often read 'arrow'.
- $\mathrm{P} \rightarrow \mathrm{Q}$ is called a *conditional.* Its *antecedent* is P and its *consequent* is Q.
- $\mathrm{P} \rightarrow \mathrm{Q}$ is false if P is true and Q is false—and true otherwise.
- By that definition, $\mathrm{P} \rightarrow \mathrm{Q}$ is equivalent to $(\neg\mathrm{P}) \lor \mathrm{Q}$
- Some authors use $\supset$ (hook).

Biconditional: $\leftrightarrow$

- Translates to English as 'if and only if'
- $\mathrm{P} \leftrightarrow \mathrm{Q}$ is called a *biconditional.*
- $\mathrm{P} \leftrightarrow \mathrm{Q}$ is true if P and Q both are true or both are false—and false otherwise.

6 `https://en.wikipedia.org/wiki/Sheffer%20stroke`

- By that definition, $P \leftrightarrow Q$ is equivalent to the more verbose $(P \land Q) \lor ((\neg P) \land (\neg Q))$. It is also equivalent to $(P \rightarrow Q) \land (Q \rightarrow P)$, the conjunction of two conditionals where in the second conditional the antecedent and consequent are reversed from the first.
- Some authors use $\equiv$.

2.1.3 Grouping

Parentheses (and) are used for grouping. Thus

$$((P \land Q) \rightarrow R)$$

$$(P \land (Q \rightarrow R))$$

are two different and distinct sentences. Each negation, conjunction, disjunction, conditional, and biconditionals gets a single pair or parentheses.

2.2 Notes

(1) An *atomic sentence* is a sentence consisting of just a single sentence letter. A *molecular sentence* is a sentence with at least one sentential connective. The *main connective* of a molecular formula is the connective that governs the entire sentence. Atomic sentences, of course, do not have a main connective.

(2) The $\supset$ and $\equiv$ signs for conditional and biconditional are historically older, perhaps a bit more traditional, and definitely occur more commonly in WikiBooks[7] and Wikipedia[8] than our arrow and double arrow. They originate with Alfred North Whitehead[9] and Bertrand Russell[10] in *Principia Mathematica*[11]. Our arrow and double arrow appear to originate with Alfred Tarski[12], and may be a bit more popular today than the Whitehead and Russell's $\supset$ and $\equiv$.

(3) Sometimes you will see people reading our arrow as *implies*. This is fairly common in WikiBooks[13] and Wikipedia[14]. However, most logicians prefer to reserve 'implies' for metalinguistic use. They will say:

If P then Q

or even

P arrow Q

7 https://en.wikibooks.org/wiki/Main_Page
8 https://en.wikipedia.org/wiki/Main_Page
9 https://en.wikipedia.org/wiki/Alfred%20North%20Whitehead
10 https://en.wikipedia.org/wiki/Bertrand%20Russell
11 https://en.wikipedia.org/wiki/Principia_Mathematica
12 https://en.wikipedia.org/wiki/Alfred%20Tarski
13 https://en.wikibooks.org/wiki/Main_Page
14 https://en.wikipedia.org/wiki/Main_Page

They approve of:

'P' implies 'Q'

but will frown on:

P implies Q

2.3 Translation

Consider the following English sentences:

If it is raining and Jones is out walking, then Jones has an umbrella.

If it is Tuesday or it is Wednesday, then Jones is out walking.

To render these in $\mathcal{L}_{\mathcal{S}}$, we first specify an appropriate English translation for some sentence letters.

P : It is raining.

Q : Jones is out walking.

R : Jones has an umbrella.

S : It is Tuesday.

T : It is Wednesday.

We can now partially translate our examples as:

$$\text{If P and Q, then R}$$

$$\text{If S or T, then Q}$$

Then finish the translation by adding the sentential connectives and parentheses:

$$((\mathrm{P} \wedge \mathrm{Q}) \rightarrow \mathrm{R})$$

$$((\mathrm{S} \vee \mathrm{T}) \rightarrow \mathrm{Q})$$

2.4 Quoting convention

For English expressions, we follow the logical tradition of using single quotes. This allows us to use ' 'It is raining' ' as a quotation of 'It is raining'.

For expressions in $\mathcal{L}_{\mathcal{S}}$, it is easier to treat them as self-quoting so that the quotation marks are implicit. Thus we say that the above example translates $\mathrm{S} \rightarrow \mathrm{P}$ (note the lack of quotes) as 'If it is Tuesday, then It is raining'.

3 Formal Syntax

In The Sentential Language[1], we informally described our sentential language. Here we give its formal syntax or grammar. We will call our language $\mathcal{L_S}$.

3.1 Vocabulary

- *Sentence letters*: Capital letters 'A' – 'Z', each with (1) a superscript '0' and (2) a natural number subscript. (The *natural numbers* are the set of positive integers and zero.) Thus the sentence letters are:

$$\mathrm{A}^0_0,\ \mathrm{A}^0_1,\ ...,\ \mathrm{B}^0_0,\ \mathrm{B}^0_1,\ ...,\ ...,\ \mathrm{Z}^0_0,\ \mathrm{Z}^0_1,\ ...$$

- *Sentential connectives*:

$$\land,\ \lor,\ \neg,\ \rightarrow,\ \leftrightarrow$$

- Grouping symbols:

$$(,\ )$$

The superscripts on sentence letters are not important until we get to the predicate logic, so we won't really worry about those here. The subscripts on sentence letters are to ensure an infinite supply of sentence letters. On the next page, we will abbreviate away most superscripts and subscripts.

3.2 Expressions

Any string of characters from the $\mathcal{L_S}$ vocabulary is an *expression* of $\mathcal{L_S}$. Some expressions are grammatically correct. Some are as incorrect in $\mathcal{L_S}$ as 'Over talks David Mary the' is in English. Still other expressions are as hopelessly ill-formed in $\mathcal{L_S}$ as 'jmr.ovn asgj as;lnre' is in English.

We call a grammatically correct expression of $\mathcal{L_S}$ a well-formed formula. When we get to Predicate Logic, we will find that only some well formed formulas are sentences. For now though, we consider every well formed formula to be a sentence.

1 Chapter 1.3.3 on page 9

3.3 Formation rules

An expression of $\mathcal{L}_\mathcal{S}$ is a *well-formed formula* of $\mathcal{L}_\mathcal{S}$ if and only if it is constructed according to the following rules.

i. A sentence letter is a well-formed formula.

ii. If φ and ψ are well-formed formulae, then so are each of:

$$\text{ii-a.}\quad \neg\varphi$$

$$\text{ii-b.}\quad (\varphi \wedge \psi)$$

$$\text{ii-c.}\quad (\varphi \vee \psi)$$

$$\text{ii-d.}\quad (\varphi \rightarrow \psi)$$

$$\text{ii-e.}\quad (\varphi \leftrightarrow \psi)$$

In general, we will use 'formula' as shorthand for 'well-formed formula'. Since all formulae in $\mathcal{L}_\mathcal{S}$ are sentences, we will use 'formula' and 'sentence' interchangeably.

3.4 Quoting convention

We will take expressions of $\mathcal{L}_\mathcal{S}$ to be self-quoting and so regard

$$(\mathrm{P}_0^0 \rightarrow \mathrm{Q}_0^0)$$

to include implicit quotation marks. However, something like

$$(1) \quad (\varphi \rightarrow \psi)$$

requires special consideration. It is not itself an expression of $\mathcal{L}_\mathcal{S}$ since φ and ψ are not in the vocabulary of $\mathcal{L}_\mathcal{S}$. Rather they are used as variables in English which range over expressions of $\mathcal{L}_\mathcal{S}$. Such a variable is called a *metavariable*, and an expression using a mix of vocabulary from $\mathcal{L}_\mathcal{S}$ and metavariables is called a *metalogical expression.* Suppose we let φ be P_0^0 and ψ be $(\mathrm{Q}_0^0 \vee \mathrm{R}_0^0)$. Then (1) becomes

$('\mathrm{P}_0^0' \rightarrow ('\mathrm{Q}_0^0' \vee '\mathrm{R}_0^0'))$

which is not what we want. Instead we take (1) to mean (using explicit quotes):

the expression consisting of '(' followed by φ followed by '$\rightarrow$' followed by ψ followed by ')' .

Explicit quotes following this convention are called *Quine quotes* or *corner quotes*. Our corner quotes will be implicit.

3.5 Additional terminology

We introduce (or, in some cases, repeat) some useful syntactic terminology.

- We distinguish between an expression (or a formula) and an *occurrence* of an expression (or formula). The formula

$$((P^0_0 \land P^0_0) \land \neg P^0_0)$$

is the same formula no matter how many times it is written. However, it contains three occurrences of the sentence letter P^0_0 and two occurrences of the sentential connective $\land$.

- ψ is a *subformula* of φ if and only if φ and ψ are both formulae and φ contains an occurrence of ψ. ψ is a *proper subformula* of φ if and only if (i) ψ is a subformula of φ and (ii) ψ is not the same formula as φ.
- An *atomic formula* or *atomic sentence* is one consisting solely of a sentence letter. Or put the other way around, it is a formula with no sentential connectives. A *molecular formula* or *molecular sentence* is one which contains at least one occurrence of a sentential connective.
- The *main connective* of a molecular formula is the last occurrence of a connective added when the formula was constructed according to the rules above.
- A *negation* is a formula of the form $\neg\varphi$ where φ is a formula.
- A *conjunction* is a formula of the form $(\varphi \land \psi)$ where φ and ψ are both formulae. In this case, φ and ψ are both *conjuncts*.
- A *disjunction* is a formula of the form $(\varphi \lor \psi)$ where φ and ψ are both formulae. In this case, φ and ψ are both *disjuncts*.
- A *conditional* is a formula of the form $(\varphi \to \psi)$ where φ and ψ are both formulae. In this case, φ is the *antecedent*, and ψ is the *consequent*. The *converse* of $(\varphi \to \psi)$ is $(\psi \to \varphi)$. The *contrapositive* of $(\varphi \to \psi)$ is $(\neg\psi \to \neg\varphi)$.
- A *biconditional* is a formula of the form $(\varphi \leftrightarrow \psi)$ where φ and ψ are both formulae.

3.6 Examples

$$(1) \quad (\neg(P^0_0 \lor Q^0_0) \to (R^0_0 \land \neg Q^0_0))$$

By rule (i), all sentence letters, including

$$P^0_0,\ Q^0_0,\ \text{and } R^0_0\ ,$$

are formulae. By rule (ii-a), then, the negation

$$\neg Q^0_0$$

is also a formula. Then by rules (ii-c) and (ii-b), we get the disjunction and conjunction

$$(P^0_0 \vee Q^0_0),\ \text{and}\ (R^0_0 \wedge \neg Q^0_0)$$

as formulae. Applying rule (ii-a) again, we get the negation

$$\neg(P^0_0 \vee Q^0_0)$$

as a formula. Finally, rule (ii-c) generates the conditional of (1), so it too is a formula.

$$(2) \quad ((P^0_0 \neg \wedge Q^0_0) \vee S^0_0)$$

This appears to be generated by rule (ii-c) from

$$(P^0_0 \neg \wedge Q^0_0),\ \text{and}\ S^0_0\ .$$

The second of these is a formula by rule (i). But what about the first? It would have to be generated by rule (ii-b) from

$$P^0_0 \neg\ \text{and}\ Q^0_0\ .$$

But

$$P^0_0 \neg$$

cannot be generated by rule (ii-a). So (2) is not a formula.

4 Informal Conventions

In The Sentential Language[1], we gave an informal description of a sentential language, namely $\mathcal{L}_{\mathcal{S}}$. We have also given a Formal Syntax[2] for $\mathcal{L}_{\mathcal{S}}$. Our official grammar generates a large number of parentheses. This makes formal definitions and other specifications easier to write, but it makes the language rather cumbersome to use. In addition, all the subscripts and superscripts quickly get to be unnecessarily tedious. The end result is an ugly and difficult to read language.

We will continue to use official grammar for specifying formalities. However, we will informally use a less cumbersome variant for other purposes. The transformation rules below convert official formulae of $\mathcal{L}_{\mathcal{S}}$ into our informal variant.

4.1 Transformation rules

We create informal variants of official $\mathcal{L}_{\mathcal{S}}$ formulae as follows. The examples are cumulative.

- The official grammar required sentence letters to have the superscript '0'. Superscripts aren't necessary or even useful until we get to the predicate logic, so we will always omit them in our informal variant. We will write, for example, P_0 instead of P^0_0.
- We will omit the subscript if it is '0'. Thus we will write P instead of P^0_0. However, we cannot omit all subscripts; we still need to write, for example, P_1.
- We will omit outermost parentheses. For example, we will write

$$\mathrm{P} \rightarrow \mathrm{Q}$$

instead of

$$(\mathrm{P}^0_0 \rightarrow \mathrm{Q}^0_0) \ .$$

- We will let a series of the same binary connective associate on the right. For example, we can transform the official

$$(\mathrm{P}^0_0 \land (\mathrm{Q}^0_0 \land \mathrm{R}^0_0))$$

into the informal

1 Chapter 1.3.3 on page 9
2 Chapter 2.4 on page 15

$$P \land Q \land R \ .$$

However, the best we can do with

$$((P_0^0 \land Q_0^0) \land R^0)$$

is

$$(P \land Q) \land R \ .$$

- We will use precedence rankings to omit internal parentheses when possible. For example, we will regard $\rightarrow$ as having lower precedence (wider scope) than $\lor$. This allows us to write

$$P \rightarrow Q \lor R$$

instead of

$$(P_0^0 \rightarrow (Q_0^0 \lor R_0^0)) \ .$$

However, we cannot remove the internal parentheses from

$$((P_0^0 \rightarrow Q_0^0) \lor R_0^0) \ .$$

Our informal variant of this latter formula is

$$(P \rightarrow Q) \lor R \ .$$

Full precedence rankings are given below.

4.2 Precedence and scope

Precedence rankings indicate the order that we evaluate the sentential connectives. $\lor$ has a higher precedence than $\rightarrow$. Thus, in calculating the truth value of

$$(1) \quad P \rightarrow Q \lor R \ ,$$

we start by evaluating the truth value of

$$(2) \quad Q \lor R$$

first. Scope is the length of expression that is governed by the connective. The occurrence of $\rightarrow$ in (1) has a wider scope than the occurrence of $\lor$. Thus the occurrence of $\rightarrow$ in (1) governs the whole sentence while the occurrence of $\lor$ in (1) governs only the occurrence of (2) in (1).

The full ranking from highest precedence (narrowest scope) to lowest precedence (widest scope) is:

$\neg$	highest precedence (narrowest scope)
$\wedge$	
$\vee$	
$\rightarrow$	
$\leftrightarrow$	lowest precedence (widest scope)

4.3 Examples

Let's look at some examples. First,

$$((\mathrm{P}_0^0 \rightarrow \mathrm{Q}_0^0) \leftrightarrow ((\mathrm{P}_0^0 \wedge \mathrm{Q}_0^0) \vee ((\neg \mathrm{P}_0^0 \wedge \mathrm{Q}_0^0) \vee (\neg \mathrm{P}_0^0 \wedge \neg \mathrm{Q}_0^0))))$$

can be written informally as

$$\mathrm{P} \rightarrow \mathrm{Q} \leftrightarrow \mathrm{P} \wedge \mathrm{Q} \vee \neg \mathrm{P} \wedge \mathrm{Q} \vee \neg \mathrm{P} \wedge \neg \mathrm{Q} \ .$$

Second,

$$((\mathrm{P}_0^0 \leftrightarrow \mathrm{Q}_0^0) \leftrightarrow ((\mathrm{P}_0^0 \wedge \mathrm{Q}_0^0) \vee (\neg \mathrm{P}_0^0 \wedge \neg \mathrm{Q}_0^0)))$$

can be written informally as

$$(\mathrm{P} \leftrightarrow \mathrm{Q}) \leftrightarrow \mathrm{P} \wedge \mathrm{Q} \vee \neg \mathrm{P} \wedge \neg \mathrm{Q} \ .$$

Some unnecessary parentheses may prove helpful. In the two examples above, the informal variants may be easier to read as

$$(\mathrm{P} \rightarrow \mathrm{Q}) \leftrightarrow (\mathrm{P} \wedge \mathrm{Q}) \vee (\neg \mathrm{P} \wedge \mathrm{Q}) \vee (\neg \mathrm{P} \wedge \neg \mathrm{Q})$$

and

$$(\mathrm{P} \leftrightarrow \mathrm{Q}) \leftrightarrow (\mathrm{P} \wedge \mathrm{Q}) \vee (\neg \mathrm{P} \wedge \neg \mathrm{Q}) \ .$$

Note that the informal formula

$$\neg \mathrm{P} \rightarrow \mathrm{Q}$$

is restored to its official form as

$$(\neg \mathrm{P}_0^0 \rightarrow \mathrm{Q}_0^0) \ .$$

By contrast, the informal formula

$$\neg(\mathrm{P} \to \mathrm{Q})$$

is restored to its official form as

$$\neg(\mathrm{P}^0_0 \to \mathrm{Q}^0_0)\ .$$

5 Formal Semantics

English syntax for 'Dogs bark' specifies that it consists of a plural noun followed by an intransitive verb. English semantics for 'Dogs bark' specify its meaning, namely that dogs bark.

In The Sentential Language[1], we gave an informal description of $\mathcal{L}_\mathcal{S}$. We also gave a Formal Syntax[2]. However, at this point our language is just a toy, a collection of symbols we can string together like beads on a necklace. We do have rules for how those symbols are to be ordered. But at this point those might as well be aesthetic rules. The difference between well-formed formulae and ill-formed expressions is not yet any more significant that the difference between pretty and ugly necklaces. In order for our language to have any meaning, to be usable in saying things, we need a formal semantics.

Any given formal language can be paired with any of a number of competing semantic rule sets. The semantics we define here is the usual one for modern logic. However, alternative semantic rule-sets have been proposed. Alternative semantic rule-sets of $\mathcal{L}_\mathcal{S}$ have included (but are certainly not limited to) intuitionistic logics[3], relevance logics[4], non-monotonic logics[5], and multi-valued logics[6].

5.1 Formal semantics

The formal semantics for a formal language such as $\mathcal{L}_\mathcal{S}$ goes in two parts.

- Rules for specifying an interpretation. An *interpretation* assigns semantic values to the non-logical symbols of a formal syntax. The semantics for a formal language will specify what range of values can be assigned to which class of non-logical symbols. $\mathcal{L}_\mathcal{S}$ has only one class of non-logical symbols, so the rule here is particularly simple. An interpretation for a sentential language is a *valuation*, namely an assignment of truth values to sentence letters. In predicate logic, we will encounter interpretations that include other elements in addition to a valuation.
- Rules for assigning semantic values to larger expressions of the language. For sentential logic, these rules assign a truth value to larger formulae based on truth values assigned to smaller formulae. For more complex syntaxes (such as for predicate logic), values are assigned in a more complex fashion.

1 Chapter 1.3.3 on page 9
2 Chapter 2.4 on page 15
3 `https://en.wikipedia.org/wiki/Intuitionistic%20logic%20`
4 `https://en.wikipedia.org/wiki/Relevance%20logic%20`
5 `https://en.wikipedia.org/wiki/Non-monotonic%20logic%20`
6 `https://en.wikipedia.org/wiki/Multi-valued%20logic%20`

An *extended valuation* assigns truth values to the molecular formulae of $\mathcal{L}_{\mathcal{S}}$ (or similar sentential language) based on a valuation. A valuation for sentence letters is extended by a set of rules to cover all formulae.

5.2 Valuations

We can give a (partial) valuation $\mathfrak{v}$ as:

$$\mathrm{P}_0 \ : \ \text{True}$$

$$\mathrm{P}_1 \ : \ \text{False}$$

$$\mathrm{P}_2 \ : \ \text{False}$$

$$\mathrm{P}_3 \ : \ \text{False}$$

(Remember that we are abbreviating our sentence letters by omitting superscripts.)

Usually, we are only interested in the truth values of a few sentence letters. The truth values assigned to other sentence letters can be random.

Given this valuation, we say:

$$\mathfrak{v}[\mathrm{P}_0] \ = \ \text{True}$$

$$\mathfrak{v}[\mathrm{P}_1] \ = \ \text{False}$$

$$\mathfrak{v}[\mathrm{P}_2] \ = \ \text{False}$$

$$\mathfrak{v}[\mathrm{P}_3] \ = \ \text{False}$$

Indeed, we can define a valuation as a function taking sentence letters as its arguments and truth values as its values (hence the name 'truth value'). Note that $\mathcal{L}_{\mathcal{S}}$ does not have a fixed interpretation or valuation for sentence letters. Rather, we specify interpretations for temporary use.

5.3 Extended valuations

An extended interpretation generates the truth values of longer sentences given an interpretation. For sentential logic, an interpretation is a valuation, so an extended interpretation is an extended valuation. We define an extension $\overline{\mathfrak{v}}$ of valuation $\mathfrak{v}$ as follows.

i. For π a sentence letter, $\overline{\mathfrak{v}}[\pi] = \mathfrak{v}[\pi]$.

$$\text{ii.} \quad \overline{\mathfrak{v}}[\neg\varphi] = \begin{cases} \text{True} & \text{if } \overline{\mathfrak{v}}[\varphi] = \text{False;} \\ \text{False} & \text{otherwise (i.e., if } \overline{\mathfrak{v}}[\varphi] = \text{True).} \end{cases}$$

$$\text{iii.} \quad \overline{\mathfrak{v}}[(\varphi \wedge \psi)] = \begin{cases} \text{True} & \text{if } \overline{\mathfrak{v}}[\varphi] = \overline{\mathfrak{v}}[\psi] = \text{True;} \\ \text{False} & \text{otherwise.} \end{cases}$$

$$\text{iv.} \quad \overline{\mathfrak{v}}[(\varphi \vee \psi)] = \begin{cases} \text{True} & \text{if } \overline{\mathfrak{v}}[\varphi] = \text{True or } \overline{\mathfrak{v}}[\psi] = \text{True (or both);} \\ \text{False} & \text{otherwise.} \end{cases}$$

$$\text{v.} \quad \overline{\mathfrak{v}}[(\varphi \to \psi)] = \begin{cases} \text{True} & \text{if } \overline{\mathfrak{v}}[\varphi] = \text{False} \;\text{ or }\; \overline{\mathfrak{v}}[\psi] = \text{True} \;\text{ (or both);} \\ \text{False} & \text{otherwise.} \end{cases}$$

$$\text{vi.} \quad \overline{\mathfrak{v}}[(\varphi \leftrightarrow \psi)] = \begin{cases} \text{True} & \text{if } \overline{\mathfrak{v}}[\varphi] = \overline{\mathfrak{v}}[\psi]; \\ \text{False} & \text{otherwise.} \end{cases}$$

5.4 Example

We will determine the truth value of this example sentence given two valuations.

$$(1) \quad \mathrm{P} \wedge \mathrm{Q} \to \neg(\mathrm{Q} \vee \mathrm{R})$$

First, consider the following valuation:

$$\mathrm{P} : \text{True}$$

$$\mathrm{Q} : \text{True}$$

$$\mathrm{R} : \text{False}$$

(2) By clause (i):

$$\overline{\mathfrak{v}}[\mathrm{P}] = \text{True}$$

$$\overline{\mathfrak{v}}[\mathrm{Q}] \;=\; \mathrm{True}$$

$$\overline{\mathfrak{v}}[\mathrm{R}] \;=\; \mathrm{False}$$

(3) By (1) and clause (iii),

$$\overline{\mathfrak{v}}[\mathrm{P} \wedge \mathrm{Q}] \;=\; \mathrm{True.}$$

(4) By (1) and clause (iv),

$$\overline{\mathfrak{v}}[\mathrm{Q} \vee \mathrm{R}] \;=\; \mathrm{True.}$$

(5) By (4) and clause (v),

$$\overline{\mathfrak{v}}[\neg(\mathrm{Q} \vee \mathrm{R})] \;=\; \mathrm{False.}$$

(6) By (3), (5) and clause (v),

$$\overline{\mathfrak{v}}[\mathrm{P} \wedge \mathrm{Q} \rightarrow \neg(\mathrm{Q} \vee \mathrm{R})] \;=\; \mathrm{False.}$$

Thus (1) is false in our interpretation.

Next, try the valuation:

$$\mathrm{P} \;:\; \mathrm{True}$$

$$\mathrm{Q} \;:\; \mathrm{False}$$

$$\mathrm{R} \;:\; \mathrm{True}$$

(7) By clause (i):

$$\overline{\mathfrak{v}}[\mathrm{P}] \;=\; \mathrm{True}$$

$$\overline{\mathfrak{v}}[\mathrm{Q}] \;=\; \mathrm{False}$$

$$\overline{\mathfrak{v}}[\mathrm{R}] \;=\; \mathrm{True}$$

(8) By (7) and clause (iii),

$$\overline{\mathfrak{v}}[\mathrm{P} \wedge \mathrm{Q}] \;=\; \text{False.}$$

(9) By (7) and clause (iv),

$$\overline{\mathfrak{v}}[\mathrm{Q} \vee \mathrm{R}] \;=\; \text{True.}$$

(10) By (9) and clause (v),

$$\overline{\mathfrak{v}}[\neg(\mathrm{Q} \vee \mathrm{R})] \;=\; \text{False.}$$

(11) By (8), (10) and clause (v),

$$\overline{\mathfrak{v}}[\mathrm{P} \wedge \mathrm{Q} \to \neg(\mathrm{Q} \vee \mathrm{R})] \;=\; \text{True.}$$

Thus (1) is true in this second interpretation. Note that we did a bit more work this time than necessary. By clause (v), (8) is sufficient for the truth of (1).

6 Truth Tables

In the Formal Syntax[1], we earlier gave a formal semantics for sentential logic. A *truth table* is a device for using this form syntax in calculating the truth value of a larger formula given an interpretation (an assignment of truth values to sentence letters). Truth tables may also help clarify the material from the Formal Syntax[2].

6.1 Basic tables

6.1.1 Negation

We begin with the truth table for negation. It corresponds to clause (ii) of our definition for extended valuations.

1 Chapter 2.4 on page 15
2 Chapter 2.4 on page 15

P	¬P
T	F
F	T

T and F represent *True* and *False* respectively. Each row represents an interpretation. The first column shows what truth value the interpretation assigns to the sentence letter P. In the first row, the interpretation assigns P the value True. In the second row, the interpretation assigns P the value False.

The second column shows the value $\neg$P receives under a given row's interpretation. Under the interpretation of the first row, $\neg$P has the value False. Under the interpretation of the second row, $\neg$P has the value True.

We can put this more formally. The first row of the truth table above shows that when $\mathfrak{v}$[P] = True, $\overline{\mathfrak{v}}$[$\neg$P] = False. The second row shows that when $\mathfrak{v}$[P] = False, $\overline{\mathfrak{v}}$[$\neg$P] = True. We can also put things more simply: a negation has the opposite truth value than that which is negated.

6.1.2 Conjunction

The truth table for conjunction corresponds to clause (iii) of our definition for extended valuations.

P	Q	$P \wedge Q$
T	T	T
T	F	F
F	T	F
F	F	F

Here we have two sentence letters and so four possible interpretations, each represented by a single row. The first two columns show what the four interpretations assign to P and Q. The interpretation represented by the first row assigns both sentence letters the value True, and so on. The last column shows the value assigned to $\mathrm{P} \wedge \mathrm{Q}$. You can see that the conjunction is true when both conjuncts are true—and the conjunction is false otherwise, namely when at least one conjunct is false.

6.1.3 Disjunction

The truth table for disjunction corresponds to clause (iv) of our definition for extended valuations.

P	Q	$P \vee Q$
T	T	T
T	F	T
F	T	T
F	F	F

Here we see that a disjunction is true when at least one of the disjuncts is true—and the disjunction is false otherwise, namely when both disjuncts are false.

6.1.4 Conditional

The truth table for conditional corresponds to clause (v) of our definition for extended valuations.

P	Q	$P \to Q$
T	T	T
T	F	F
F	T	T
F	F	T

A conditional is true when either its antecedent is false or its consequent is true (or both). It is false otherwise, namely when the antecedent is true and the consequent is false.

6.1.5 Biconditional

The truth table for biconditional corresponds to clause (vi) of our definition for extended valuations.

P	Q	P ↔ Q
T	T	T
T	F	F
F	T	F
F	F	T

A biconditional is true when both parts have the same truth value. It is false when the two parts have opposite truth values.

6.2 Example

We will use the same example sentence from Formal Semantics[3]:

$$\mathrm{P} \wedge \mathrm{Q} \rightarrow \neg(\mathrm{Q} \vee \mathrm{R}) \ .$$

We construct its truth table as follows:

3 Chapter 5.4 on page 27

P	Q	R	$P \wedge Q$	$Q \vee R$	$\neg(Q \vee R)$	$(P \wedge Q) \rightarrow \neg(Q \vee R)$
T	T	T	T	T	F	F
T	T	F	T	T	F	F
T	F	T	F	T	F	T
T	F	F	F	F	T	T
F	T	T	F	T	F	T
F	T	F	F	T	F	T
F	F	T	F	T	F	T
F	F	F	F	F	T	T

With three sentence letters, we need eight valuations (and so lines of the truth table) to cover all cases. The table builds the example sentence in parts. The $P \wedge Q$ column was based on the P and Q columns. The $Q \vee R$ column was based on the Q and R columns. This in turn was the basis for its negation in the next column. Finally, the last column was based on the $P \wedge Q$ and $\neg(Q \vee R)$ columns.

We see from this truth table that the example sentence is false when both P and Q are true, and it is true otherwise.

This table can be written in a more compressed format as follows.

				(1)		(4)	(3)		(2)	
P	Q	R	P	$\wedge$	Q	$\rightarrow$	$\neg$	(Q	$\vee$	R)
T	T	T		T		F	F		T	
T	T	F		T		F	F		T	
T	F	T		F		T	F		T	
T	F	F		F		T	T		F	
F	T	T		F		T	F		T	
F	T	F		F		T	F		T	
F	F	T		F		T	F		T	
F	F	F		F		T	T		F	

The numbers above the connectives are not part of the truth table but rather show what order the columns were filled in.

6.3 Satisfaction and validity of formulae

6.3.1 Satisfaction

In sentential logic, an interpretation under which a formula is true is said to *satisfy* that formula. In predicate logic, the notion of satisfaction is a bit more complex. A formula is *satisfiable* if and only if it is true under at least one interpretation (that is, if and only if at least one interpretation satisfies the formula). The example truth table of Truth Tables[4] showed that the following sentence is satisfiable.

$$\mathrm{P} \wedge \mathrm{Q} \rightarrow \neg(\mathrm{Q} \vee \mathrm{R})$$

For a simpler example, the formula P is satisfiable because it is true under any interpretation that assigns P the value True.

We can use the following convenient notation to say that the interpretation $\mathfrak{v}$ satisfies (or does not satisfy) φ.

$$\mathfrak{v} \models \varphi$$

$$\mathfrak{v} \not\models \varphi$$

We can extend the notion of satisfaction to sets of formulae. A set of formulae is satisfiable if and only if there is an interpretation under which every formula of the set is true (that is, the interpretation satisfies every formula of the set).

A formula is *unsatisfiable* if and only if there is no interpretation under which it is true. A trivial example is

$$\mathrm{P} \wedge \neg \mathrm{P}$$

You can easily confirm by doing a truth table that the formula is false no matter what truth value an interpretation assigns to P. We say that an unsatisfiable formula is *logically false*. One can say that an unsatisfiable formula of sentential logic (but not one of predicate logic) is tautologically false.

4 Chapter 5.4 on page 29

6.3.2 Validity

A formula is *valid* if and only if it is satisfied under every interpretation. For example,

$$\mathrm{P} \vee \neg \mathrm{P}$$

is valid. You can easily confirm by a truth table that it is true no matter what the interpretation assigns to P. We say that a vaild sentence is *logically true*. We call a valid formula of sentential logic—but not one of predicate logic—a *tautology*.

We can use the following convenient notation to say that φ is (or is not) valid.

$$\models \ \psi$$

$$\not\models \ \psi$$

6.3.3 Equivalence

Two formulae are *equivalent* if and only if they are true under exactly the same interpretations. You can easily confirm by truth table that any interpretation that satisfies $\mathrm{P} \wedge \mathrm{Q}$ also satisfies $\mathrm{Q} \wedge \mathrm{P}$. In addition, any interpretation that satisfies $\mathrm{Q} \wedge \mathrm{P}$ also satisfies $\mathrm{P} \wedge \mathrm{Q}$. Thus they are equivalent.

We can use the following convenient notation to say that φ and ψ are equivalent.

φ

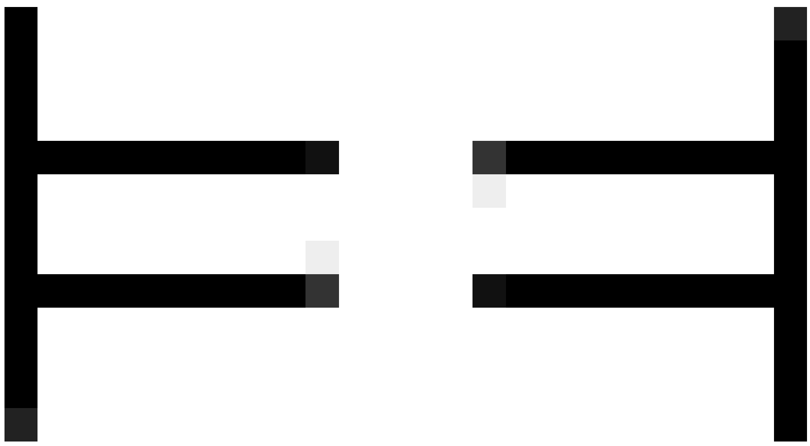

Figure 1

ψ

which is true if and only if

$$\vDash \ (\varphi \leftrightarrow \psi)$$

6.4 Validity of arguments

An *argument* is a set of formulae designated as *premises* together with a single sentence designated as the *conclusion.* Intuitively, we want the premises jointly to constitute a reason to believe the conclusion. For our purposes an argument is *any* set of premises together with *any* conclusion. That can be a bit artificial for some particularly silly arguments, but the logical properties of an argument do not depend on whether it is silly or whether anyone actually does or might consider the premises to be a reason to believe the conclusion. We consider arguments *as if* one does or might consider the premises to be a reason for the conclusion independently of whether anyone actually does or might do so. Even an empty set of premises together with a conclusion counts as an argument.

The following examples show the same argument using several notations.

Example 1

P

P → Q

Therefore Q

Example 2

P

P → Q

∴ Q

Example 3

$$\frac{\begin{array}{c}\mathrm{P}\\ \mathrm{P} \rightarrow \mathrm{Q}\end{array}}{\mathrm{Q}}$$

Example 4

$\mathrm{P},\ \mathrm{P} \rightarrow \mathrm{Q} \quad \therefore \quad \mathrm{Q}$

An argument is valid if and only if every interpretation that satisfies all the premises also satisfies the conclusion. A conclusion of a valid argument is a *logical consequence* of its premises. We can express the validity (or invalidity) of the argument with Γ as its set of premises and ψ as its conclusion using the following notation.

(1) $\Gamma \vDash \psi$

(2) $\Gamma \nvDash \psi$

For example, we have

$$\{P,\ P \to Q\} \vDash Q$$

Validity for arguments, or logical consequence, is the central notion driving the intuitions on which we build a logic. We want to know whether our arguments are good arguments, that is, whether they represent good reasoning. We want to know whether the premises of an argument constitute good reason to believe the conclusion. Validity is one essential feature of a good argument. It is not the only essential feature. A valid argument with at least one false premise is useless. Validity is the truth-preserving feature. It does not tell us that the conclusion is true, only that the logical features of the argument are such that, if the premises are true, then the conclusion is. A valid argument with true premises is *sound*.

There are other less formal features that a good argument needs. Just because the premises are true does not mean that they are believed, that we have any reason to believe them, or that we could collect evidence for them. It should also be noted that validity only applies to certain types of arguments, particularly *deductive* arguments. Deductive arguments are intended to be valid. The archetypical example for a deductive argument is a mathematical proof. Inductive arguments, of which scientific arguments provide the archetypical example, are not intended to be valid. The truth of the premises are not intended to guarantee that the conclusion is true. Rather, the truth of the premises are intended to make the truth of the conclusion highly probably or likely. In science, we do not intend to offer mathematical proofs. Rather, we gather evidence.

6.5 Formulae and arguments

For every valid formula, there is a corresponding valid argument having the valid formula as its conclusion and the empty set as its set of premises. Thus

$$\vDash \psi$$

if and only if

$$\varnothing \vDash \psi$$

For every valid argument with finitely many premises, there is a corresponding valid formula. Consider a valid argument with ψ as the conclusion and having as its premises $\varphi_1, \varphi_2, ..., \varphi_n$. Then

$$\varphi_1,\ \varphi_2,\ ...,\ \varphi_n \vDash \psi$$

There is then the corresponding valid formula

$$\varphi_1 \wedge \varphi_2 \wedge \ldots \wedge \varphi_n \to \psi$$

There corresponds to the valid argument

$\mathrm{P},\ \mathrm{P} \to \mathrm{Q} \quad \therefore \quad \mathrm{Q}$

the following valid formula:

$$\mathrm{P} \wedge (\mathrm{P} \to \mathrm{Q}) \to \mathrm{Q}$$

6.6 Implication

You may see some text reading our arrow $\to$ as 'implies' and using 'implications' as an alternative for 'conditional'. This is generally decried as a use-mention error. In ordinary English, the following are considered grammatically correct:

(3) 'That there is smoke implies that there is fire'.

(4) 'There is smoke' implies 'there is fire'.

In (3), we have one fact or proposition or whatever (the current favorite among philosophers appears to be proposition) implying another of the same species. In (4), we have one sentence implying another.

But the following is considered incorrect:

There is smoke implies there is fire.

Here, in contrast to (3), there are no quotation marks. Nothing is the subject doing the implying and nothing is the object implied. Rather, we are composing a larger sentence out of smaller ones as if 'implies' were a grammatical conjunction such as 'only if'.

Thus logicians tend to avoid using 'implication' to mean *conditional*. Rather, they use 'implies' to mean *has as a logical consequence* and 'implication' to mean *valid argument*. In doing this, they are following the model of (4) rather than (3). In particular, they read (1) and (2) as 'Γ implies (or does not imply) ψ.

7 Expressibility

7.1 Truth functions

A formula with n sentence letters requires 2^n lines in its truth table. And there are 2^m possible truth functions having a truth table of m lines. Thus there are 2^{2^n} possible truth functions of n sentence letters. There are 4 possible truth functions of one sentence letter (requiring a 2 line truth table) and 16 possible truth functions of two sentence letters (requiring a 4 line truth table). We illustrate this with the following tables. The numbered columns represent the different possibilities for the column of a main connective.

P	(i)	(ii)	(iii)	(iv)
T	T	T	F	F
F	T	F	T	F

You will recognize column (iii) as the negation truth function.

P	Q	(i)	(ii)	(iii)	(iv)	(v)	(vi)	(vii)	(viii)	(ix)	(x)	(xi)	(xii)	(xiii)	(xiv)	(xv)	(xvi)
T	T	T	T	T	T	T	T	T	T	F	F	F	F	F	F	F	F
T	F	T	T	T	T	F	F	F	F	T	T	T	T	F	F	F	F
F	T	T	T	F	F	T	T	F	F	T	T	F	F	T	T	F	F
F	F	T	F	T	F	T	F	T	F	T	F	T	F	T	F	T	F

Column (ii) represents the truth function for disjunction, column (v) represents conditional, column (vii) represents biconditional, and column (viii) represents conjunction.

7.2 Expressing arbitrary truth functions

The question arises whether we have enough connectives to represent all the truth functions of any number of sentence letters. Remember that each row represents one valuation. We can express that valuation by conjoining sentence letters assigned True under that valuation and negations of sentence letters assigned false under that valuation. The four valuations of the second table above can be expressed as

$$\mathrm{P} \wedge \mathrm{Q}$$

$$\mathrm{P} \wedge \neg\mathrm{Q}$$

$$\neg\mathrm{P} \wedge \mathrm{Q}$$

$$\neg\mathrm{P} \wedge \neg\mathrm{Q}$$

Now we can express an arbitrary truth function by disjoining the valuations under which the truth function has the value true. For example, we can express column (x) with:

(1) $(\mathrm{P} \wedge \neg\mathrm{Q}) \vee (\neg\mathrm{P} \wedge \mathrm{Q})$

You can confirm by completing the truth table that this produces the desired result. The formula is true when either (a) P is true and Q is false or (b) P is false and Q is true. There is an easier way to express this same truth function: $\mathrm{P} \leftrightarrow \neg\mathrm{Q}$. Coming up with a *simple* way to express an arbitrary truth function may require insight, but at least we have an automatic mechanism for finding *some* way to express it.

Now consider a second example. We want to express a truth function of P, Q, and R, and we want this to be true under (and only under) the following three valuations.

	(i)	(ii)	(iii)
P	True	False	False
Q	True	True	False
R	False	False	True

We can express the three conditions which yield true as

$$\mathrm{P} \wedge \mathrm{Q} \wedge \neg\mathrm{R}$$

$$\neg\mathrm{P} \wedge \mathrm{Q} \wedge \neg\mathrm{R}$$

$$\neg P \wedge \neg Q \wedge R$$

Now we need to say that *either* the first condition holds *or* that the second condition holds *or* that the third condition holds:

(2) $(P \wedge Q \wedge \neg R) \vee (\neg P \wedge Q \wedge \neg R) \vee (\neg P \wedge \neg Q \wedge R)$

You can verify by a truth table that it yields the desired result, that the formula is true in just the interpretation above.

This technique for expressing arbitrary truth functions does not work for truth functions evaluating to False in every interpretation. We need at least one interpretation yielding True in order to get the formula started. However, we can use any tautologically false formula to express such truth functions. $P \wedge \neg P$ will suffice.

7.3 Normal forms

A *normal form* provides a standardized rule of expression where any formula is equivalent to one which conforms to the rule. It will be useful in the following to define a *literal* as a sentence letter or its negation.

The technique for expressing arbitrary truth functions used formulae in disjunctive normal form. A formula in *disjunctive normal form* is a disjunction of conjunctions of literals. For the purposes of this definition, we countenance many-place disjunctions and conjunctions such as $\neg P \wedge Q \wedge \neg R$ or $\neg P \vee Q \vee \neg R$. Also for the purpose of this definition we countenance degenerate disjunctions and conjunctions of only one disjunct or conjunct. Thus we count P as being in disjunctive normal form. It is a degenerate (one-place) disjunction of a degenerate (one-place) conjunction. We could make it less degenerate (but more debauched) by converting it to the equivalent formula $(P \wedge P) \vee (P \wedge P)$.

There is another common normal form in sentential logic, conjunctive normal form. *Conjunctive normal form* is a conjunction of a disjunctions of literals. We can express arbitrary truth functions in conjunctive normal form. First, take the valuations for which the truth function evaluates to False. For each such valuation, form a disjunction of sentence letters the valuation assigns False together with the negations of sentence letters the valuation assign true. For the valuation

P : False

Q : True

R : False

we form the disjunction

$$\neg P \vee Q \vee \neg R$$

The conjunctive normal form expression of an arbitrary truth function is the conjunction of all such disjunctions matching the interpretations for which the truth function evaluates to false. The conjunctive normal form equivalent of (1) above is

$$(\neg P \vee Q) \wedge (P \vee \neg Q)$$

The conjunctive normal form equivalent of (2) above is

$$(\neg P \vee \neg Q \vee \neg R) \wedge (P \vee \neg Q \vee \neg R) \wedge (P \vee \neg Q \vee R) \wedge (P \vee Q \vee \neg R) \wedge (P \vee Q \vee R)$$

7.4 Interdefinability of connectives

Negation and conjunction are sufficient to express the other three connectives and indeed any arbitrary truth function.

$P \vee Q$

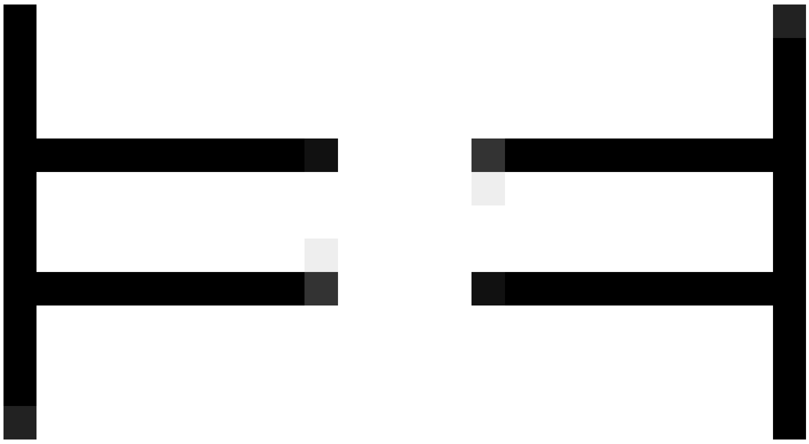

Figure 2

$\neg(\neg P \wedge \neg Q)$

$P \rightarrow Q$

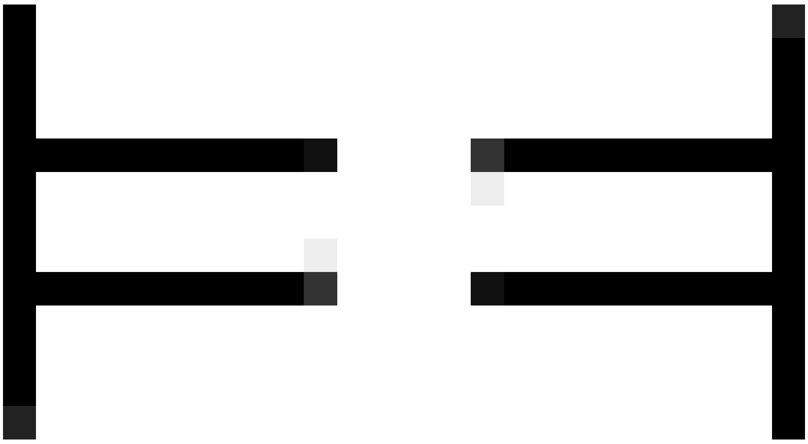

Figure 3

$\neg(P \wedge \neg Q)$

$P \leftrightarrow Q$

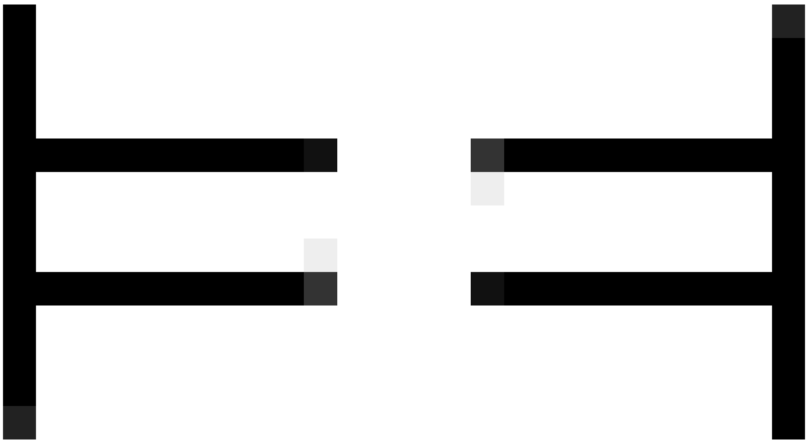

Figure 4

$(P \rightarrow Q) \wedge (Q \rightarrow P)$

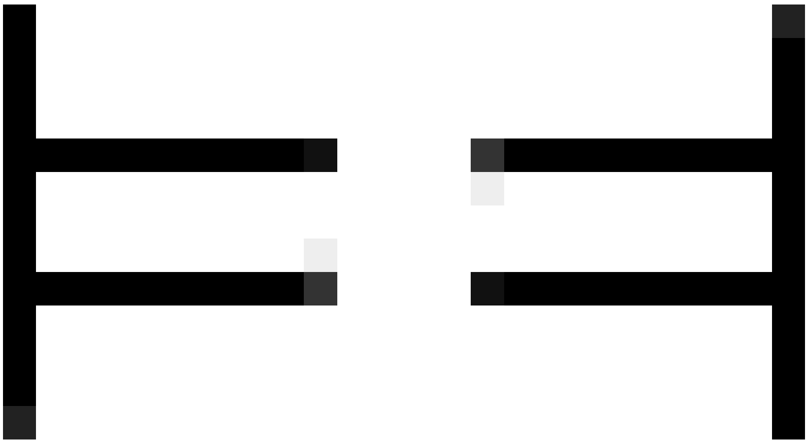

Figure 5

$\neg(P \wedge \neg Q) \wedge \neg(Q \wedge \neg P)$

Negation and disjunction are sufficient to express the other three connectives and indeed any arbitrary truth function.

$P \wedge Q$

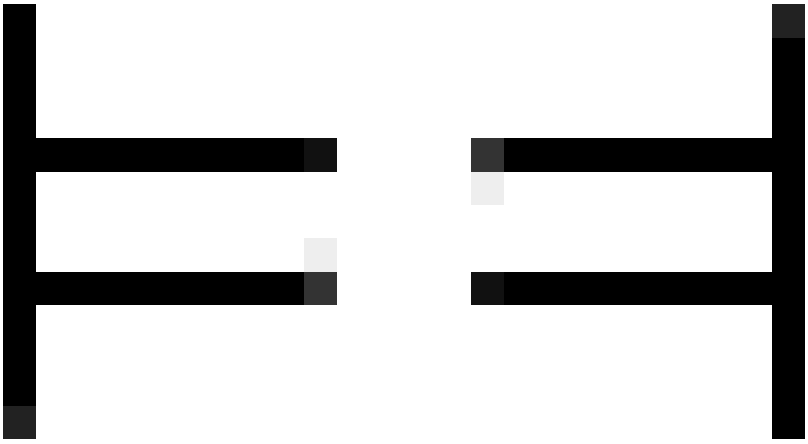

Figure 6

$\neg(\neg P \vee \neg Q)$

$P \rightarrow Q$

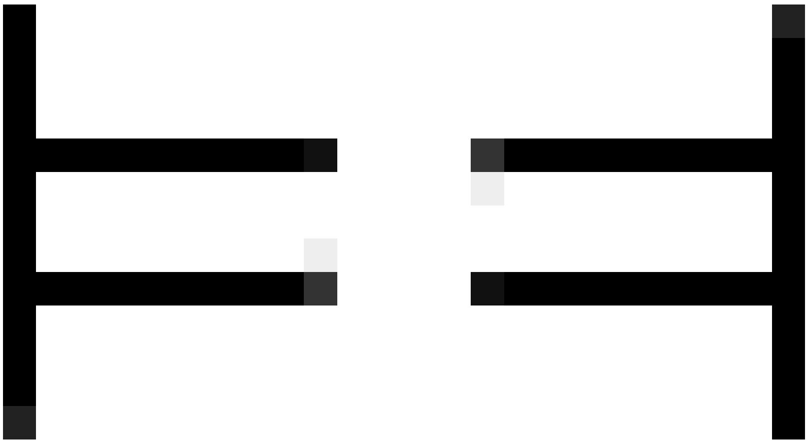

Figure 7

$\neg P \vee Q$

$P \leftrightarrow Q$

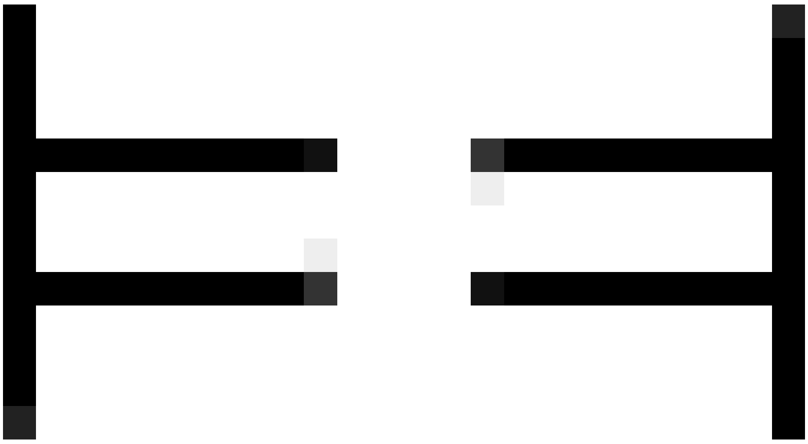

Figure 8

$(P \wedge Q) \vee (\neg P \wedge \neg Q)$

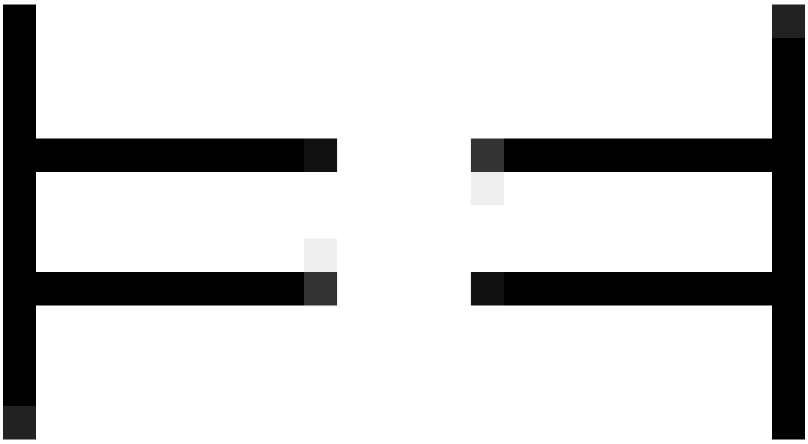

Figure 9

$\neg(\neg P \vee \neg Q) \vee \neg(P \vee Q)$

Negation and conditional are sufficient to express the other three connectives and indeed any arbitrary truth function.

$P \vee Q$

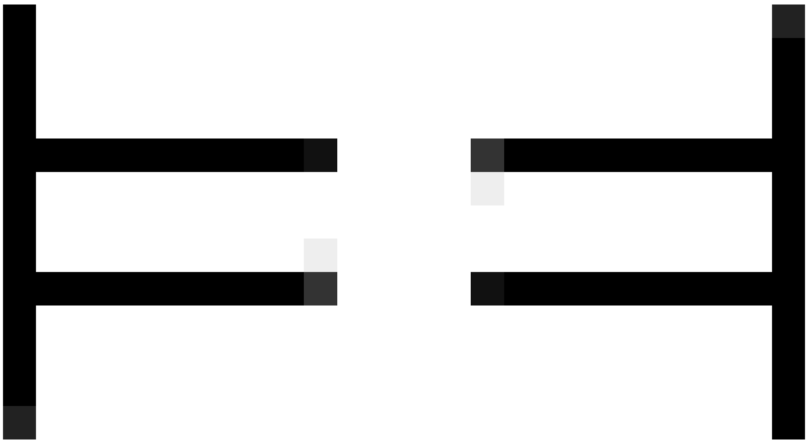

Figure 10

$\neg P \rightarrow Q$

$P \wedge Q$

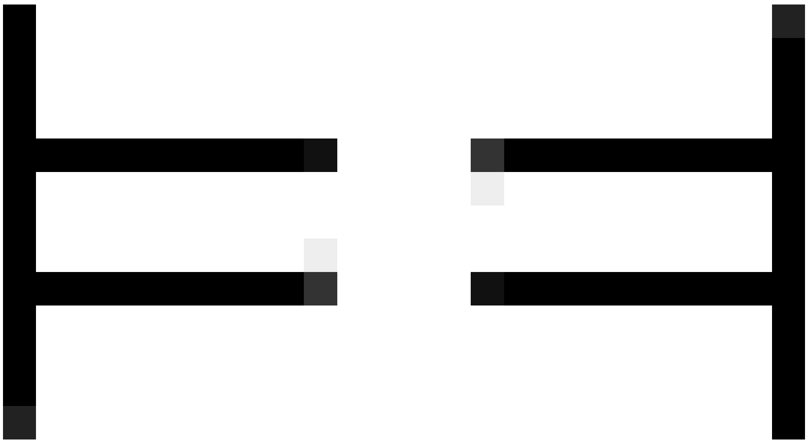

Figure 11

$\neg(P \rightarrow \neg Q)$

$P \leftrightarrow Q$

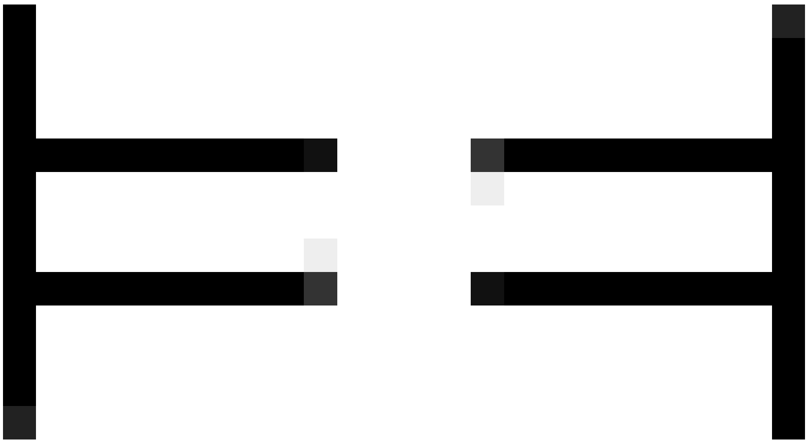

Figure 12

$(P \rightarrow Q) \wedge (Q \rightarrow P)$

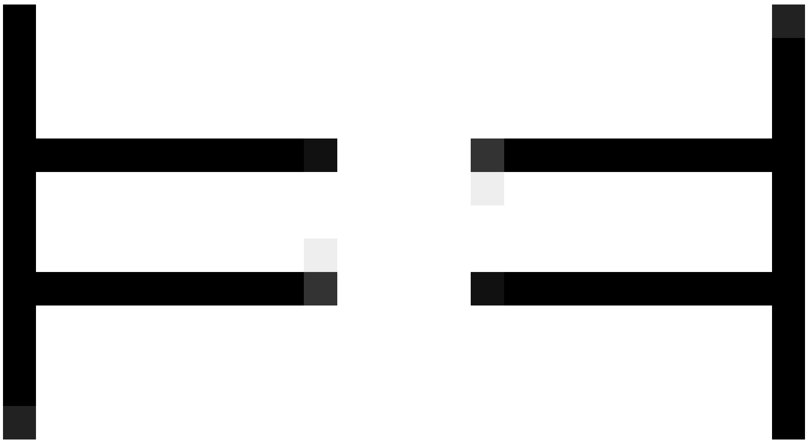

Figure 13

$\neg((\mathrm{P} \to \mathrm{Q}) \to \neg(\mathrm{Q} \to \mathrm{P}))$

Negation and biconditional are not sufficient to express the other three connectives.

7.5 Joint and alternative denials

We have seen that three pairs of connectives are each jointly sufficient to express any arbitrary truth function. The question arises, is it possible to express any arbitrary truth function with just one connective? The answer is yes, but not with any of our connectives. There are two possible binary connectives each of which, if added to $\mathcal{L}_{\mathcal{S}}$, would be sufficient.

7.5.1 Alternative denial

Alternative denial, sometimes called *NAND*. The usual symbol for this is called the *Sheffer stroke*. Temporarily add the symbol $|$ to $\mathcal{L}_{\mathcal{S}}$ and let $\mathfrak{v}[(\varphi \mid \psi)]$ be True when at least one of φ or ψ is false. It has the truth table :

P	Q	P | Q
T	T	F
T	F	T
F	T	T
F	F	T

We now have the following equivalences.

¬P

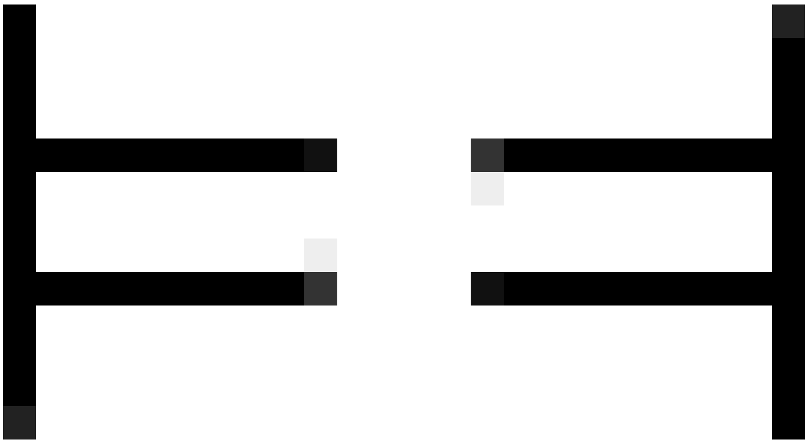

Figure 14

P | P

P ∧ Q

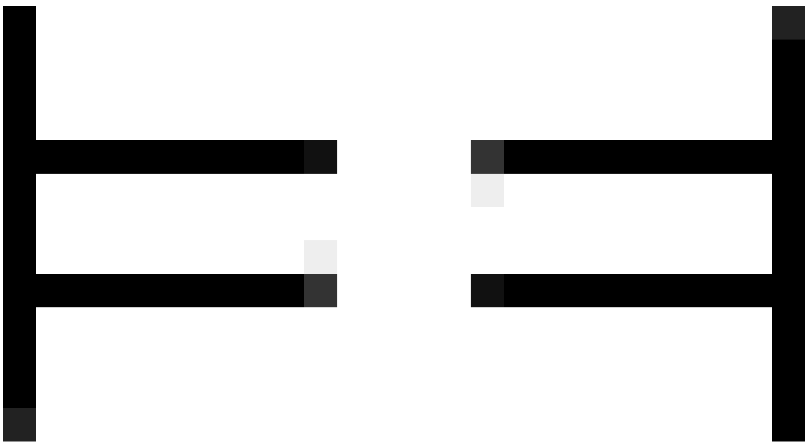

Figure 15

(P | Q) | (P | Q)

P ∨ Q

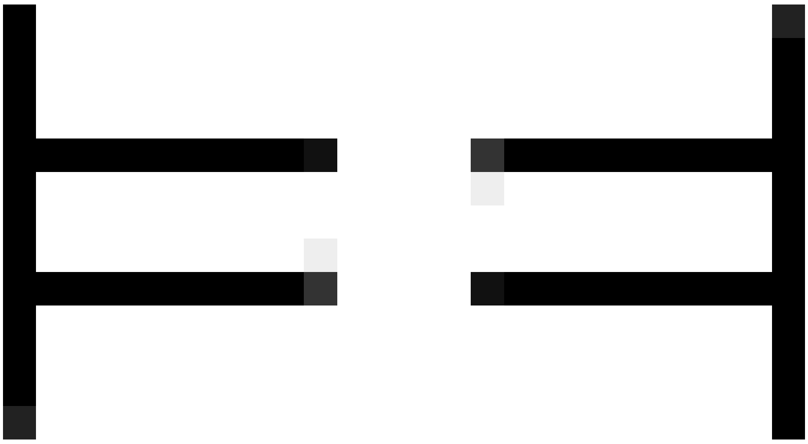

Figure 16

(P | P) | (Q | Q)

P → Q

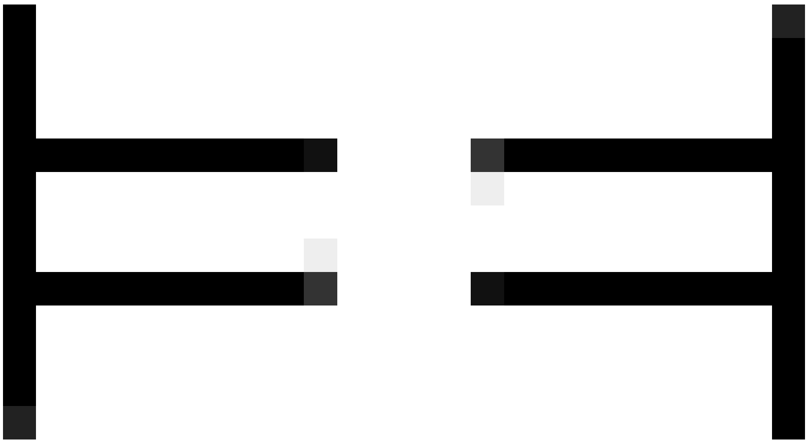

Figure 17

P | (Q | Q)

P ↔ Q

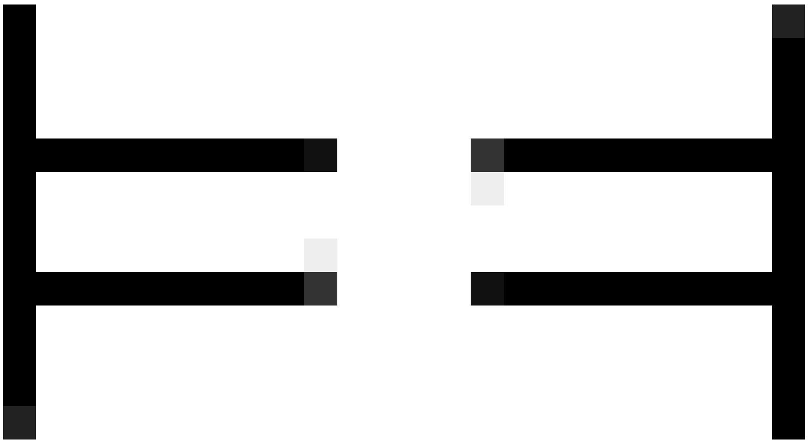

Figure 18

(P | Q) | ((P | P) | (Q | Q))

7.5.2 Joint denial

Joint denial, sometimes called *NOR*. Temporarily add the symbol $\downarrow$ to $\mathcal{L}_{\mathcal{S}}$ and let $\mathfrak{v}[(\varphi \downarrow \psi)]$ be True when both φ and ψ are false. It has the truth table :

P	Q	P ↓ Q
T	T	F
T	F	F
F	T	F
F	F	T

We now have the following equivalences.

$\neg P$

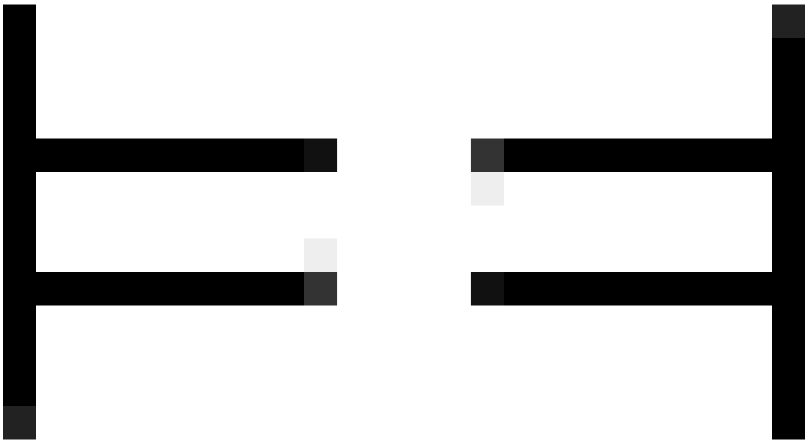

Figure 19

$P \downarrow P$

$P \wedge Q$

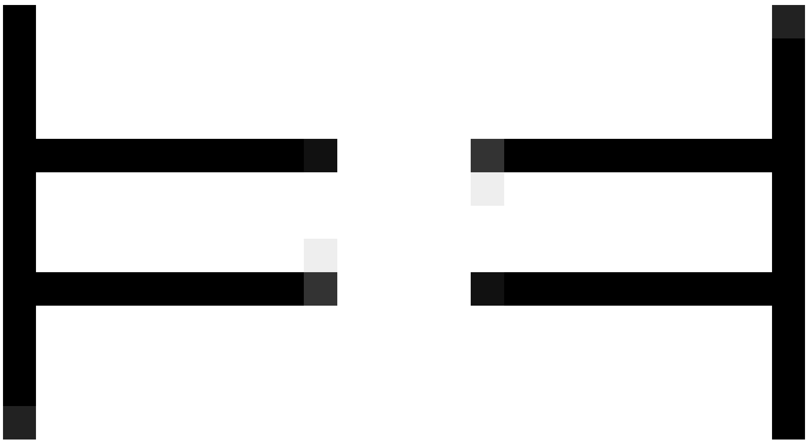

Figure 20

$(P \downarrow P) \downarrow (Q \downarrow Q)$

$P \vee Q$

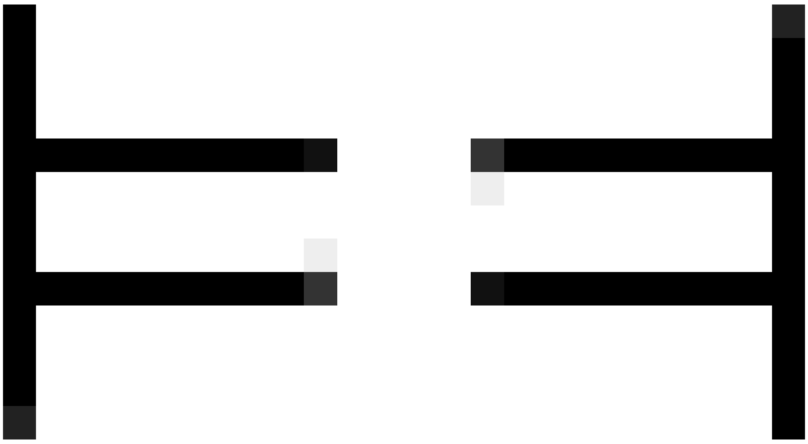

Figure 21

$(P \downarrow Q) \downarrow (P \downarrow Q)$

$P \rightarrow Q$

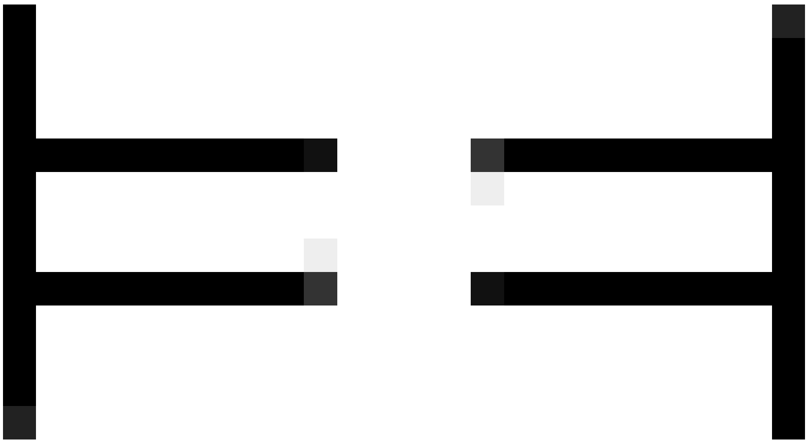

Figure 22

$((P \downarrow Q) \downarrow Q) \downarrow ((P \downarrow Q) \downarrow Q)$

$P \leftrightarrow Q$

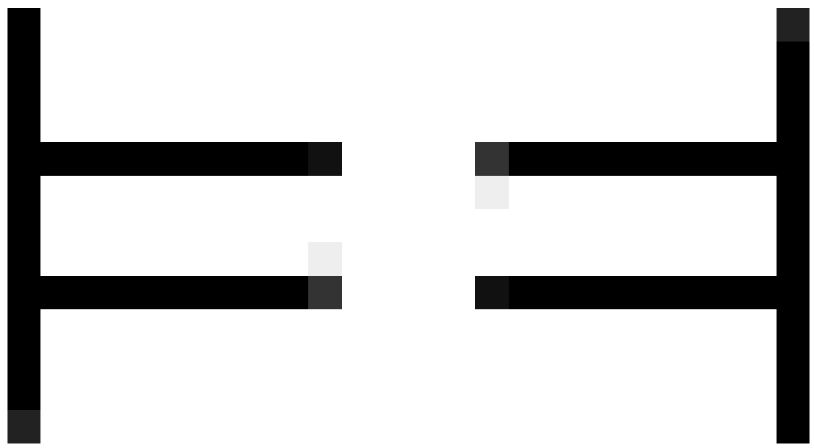

Figure 23

$((P \downarrow P) \downarrow Q) \downarrow (P \downarrow (Q \downarrow Q))$

8 Properties of Sentential Connectives

Here we list some of the more famous, historically important, or otherwise useful equivalences and tautologies. They can be added to the ones listed in Interdefinability of connectives[1]. We can go on at quite some length here, but will try to keep the list somewhat restrained. Remember that for every equivalence of φ and ψ, there is a related tautology $\varphi \leftrightarrow \psi$.

8.1 Bivalence

Every formula has exactly one of two truth values.

$\vDash$ $\mathrm{P} \lor \neg\mathrm{P}$ *Law of Excluded Middle*

$\vDash$ $\neg(\mathrm{P} \land \neg\mathrm{P})$ *Law of Non-Contradiction*

8.2 Analogues to arithmetic laws

Some familiar laws from arithmetic have analogues in sentential logic.

8.2.1 Reflexivity

Conditional and biconditional (but not conjunction and disjunction) are reflexive.

$$\vDash \ \mathrm{P} \rightarrow \mathrm{P}$$

$$\vDash \ \mathrm{P} \leftrightarrow \mathrm{P}$$

8.2.2 Commutativity

Conjunction, disjunction, and biconditional (but not conditional) are commutative.

$\mathrm{P} \land \mathrm{Q}$

1 Chapter 7.4 on page 57

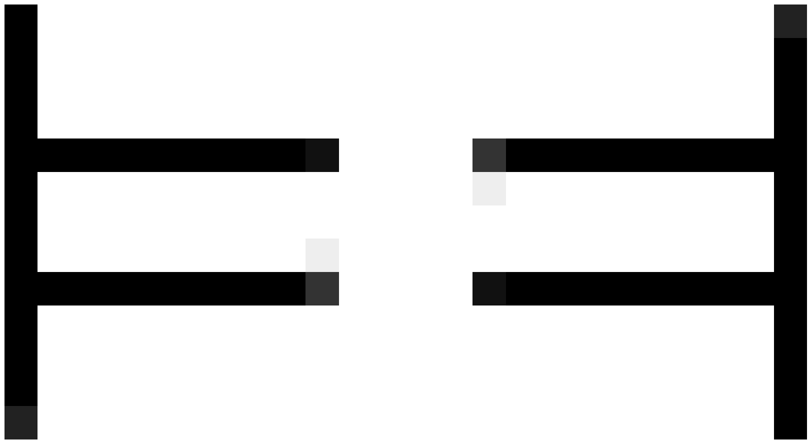

Figure 24 is equivalent to

$Q \wedge P$

$P \vee Q$

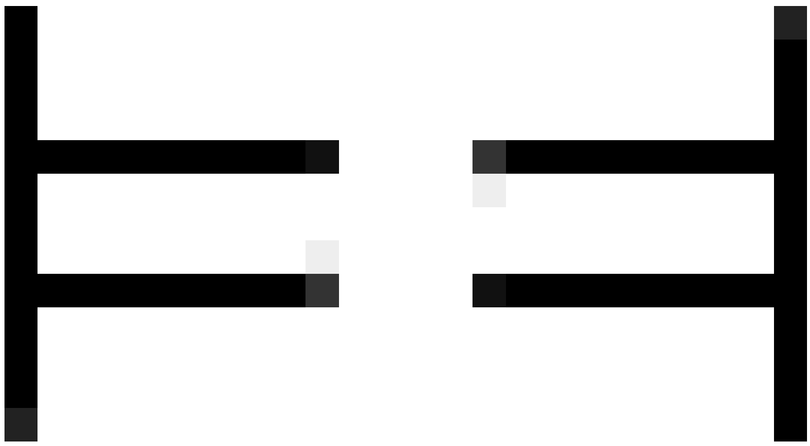

Figure 25 is equivalent to

$Q \vee P$

$P \leftrightarrow Q$

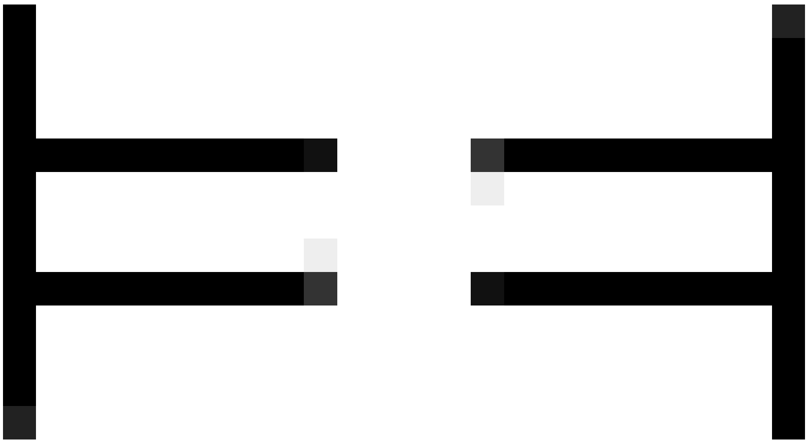

Figure 26 is equivalent to

$Q \leftrightarrow P$

8.2.3 Associativity

Conjunction, disjunction, and biconditional (but not conditional) are associative.

$(P \wedge Q) \wedge R$

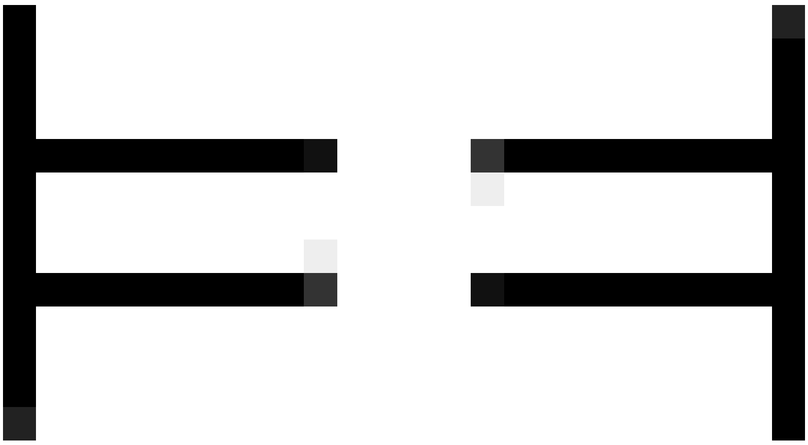

Figure 27 is equivalent to

$P \wedge (Q \wedge R)$

$(P \vee Q) \vee R$

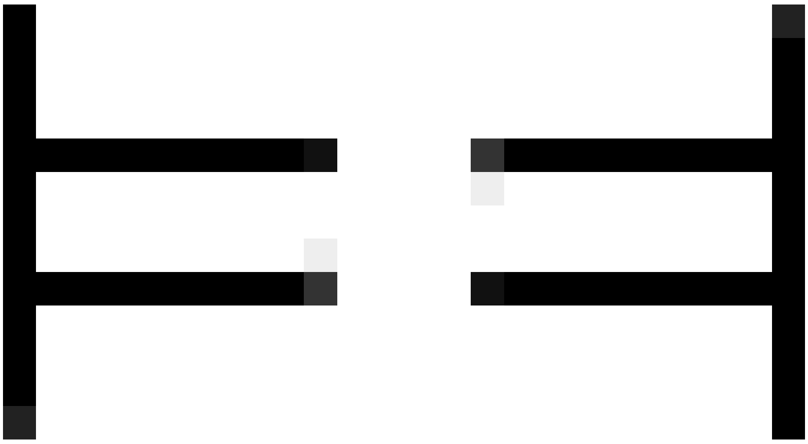

Figure 28 is equivalent to

$P \vee (Q \vee R)$

$(P \leftrightarrow Q) \leftrightarrow R$

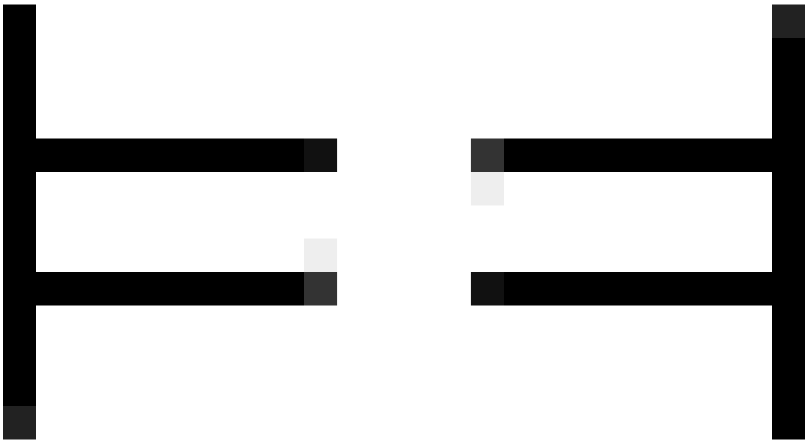

Figure 29 is equivalent to

$P \leftrightarrow (Q \leftrightarrow R)$

8.2.4 Distribution

We list ten distribution laws. Of these, probably the most important are that conjunction and disjunction distribute over each other and that conditional distributes over itself.

$P \wedge (Q \wedge R)$

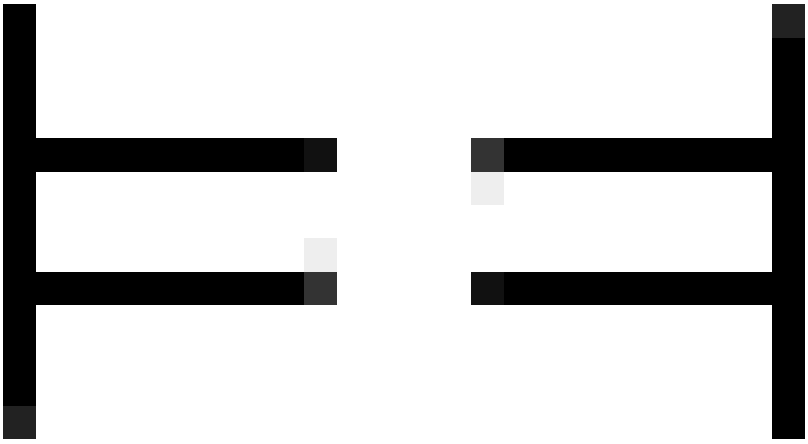

Figure 30 is equivalent to

$(P \wedge Q) \wedge (P \wedge R)$

$P \wedge (Q \vee R)$

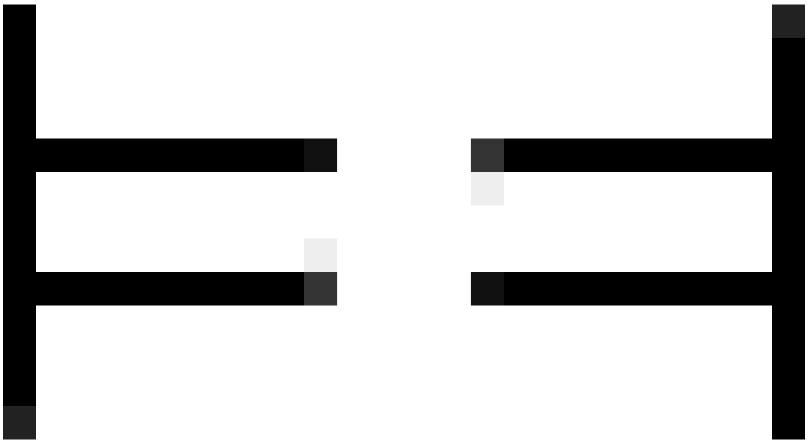

Figure 31 is equivalent to

$(P \wedge Q) \vee (P \wedge R)$

$P \vee (Q \wedge R)$

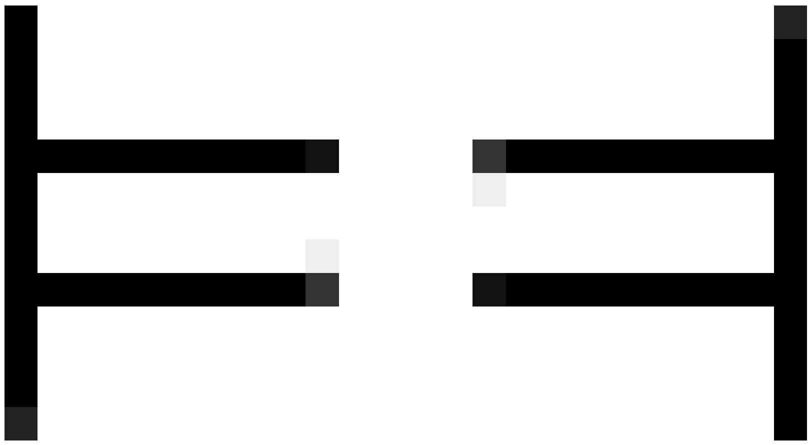

Figure 32 is equivalent to

$(P \lor Q) \land (P \lor R)$

$P \lor (Q \lor R)$

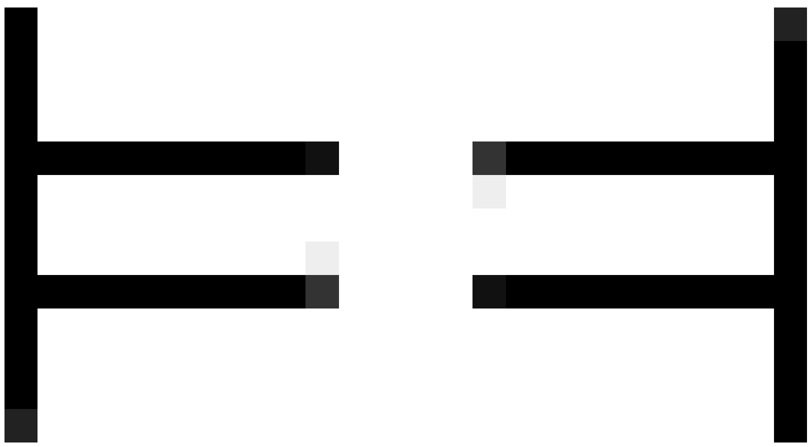

Figure 33 is equivalent to

$(P \lor Q) \lor (P \lor R)$

$P \lor (Q \to R)$

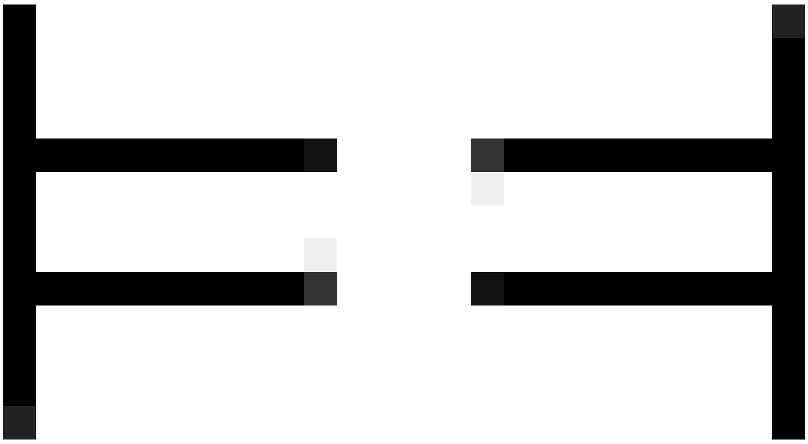

Figure 34 is equivalent to

$(P \vee Q) \rightarrow (P \vee R)$

$P \vee (Q \leftrightarrow R)$

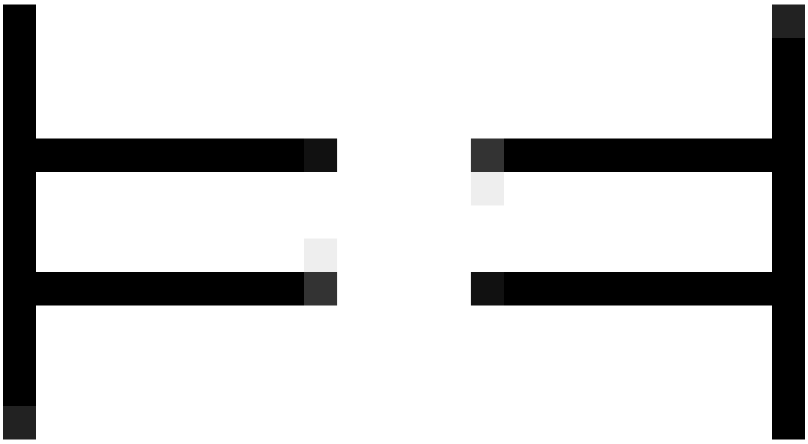

Figure 35 is equivalent to

$(P \vee Q) \leftrightarrow (P \vee R)$

$P \rightarrow (Q \wedge R)$

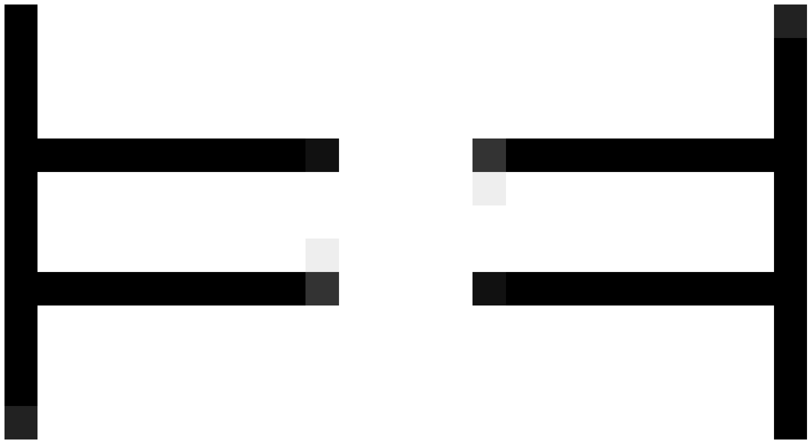

Figure 36 is equivalent to

$(P \rightarrow Q) \wedge (P \rightarrow R)$

$P \rightarrow (Q \vee R)$

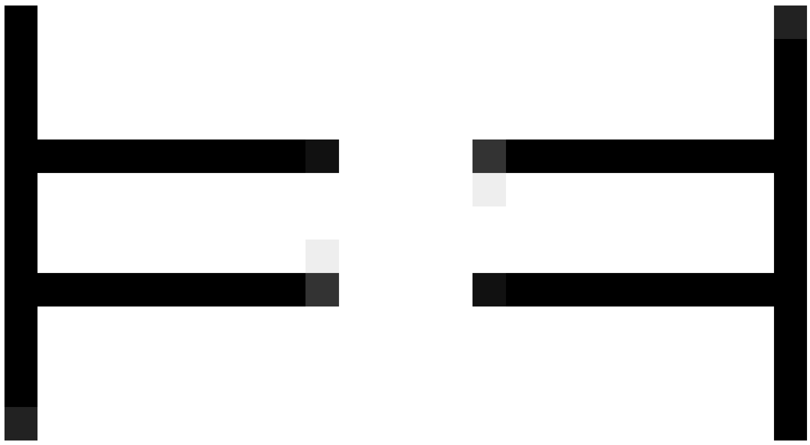

Figure 37 is equivalent to

$(P \rightarrow Q) \vee (P \rightarrow R)$

$P \rightarrow (Q \rightarrow R)$

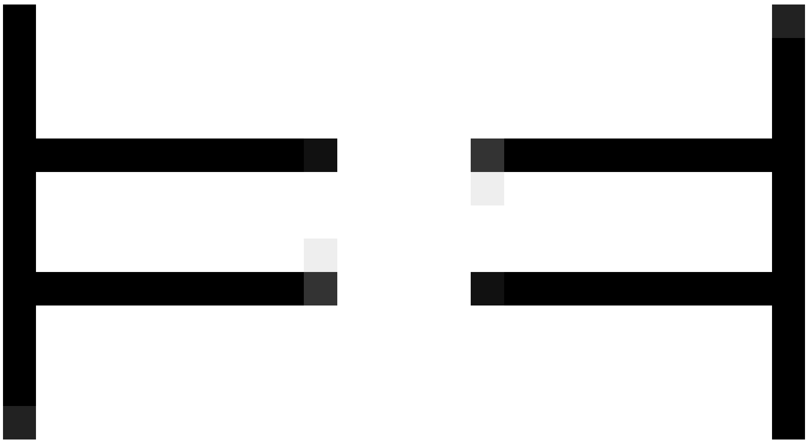

Figure 38 is equivalent to

$(P \to Q) \to (P \to R)$

$P \to (Q \leftrightarrow R)$

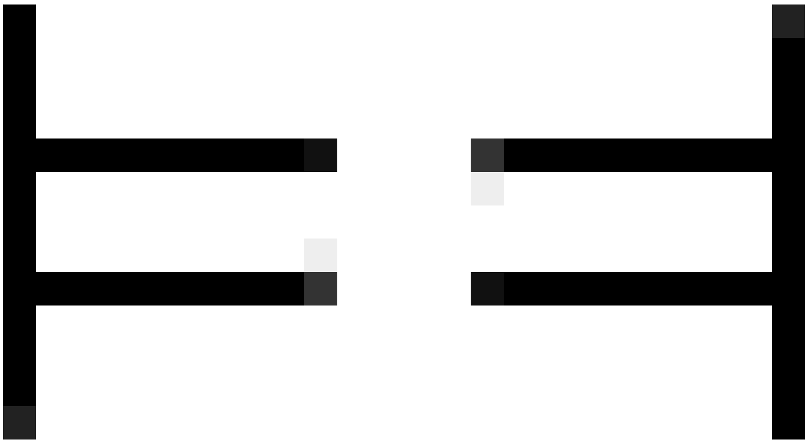

Figure 39 is equivalent to

$(P \to Q) \leftrightarrow (P \to R)$

8.2.5 Transitivity

Conjunction, conditional, and biconditional (but not disjunction) are transitive.

$$\vDash\ (P \wedge Q) \wedge (Q \wedge R) \rightarrow P \wedge R$$

$$\vDash\ (P \rightarrow Q) \wedge (Q \rightarrow R) \rightarrow (P \rightarrow R)$$

$$\vDash\ (P \leftrightarrow Q) \wedge (Q \leftrightarrow R) \rightarrow (P \leftrightarrow R)$$

8.3 Other tautologies and equivalences

8.3.1 Conditionals

These tautologies and equivalences are mostly about conditionals.

$\vDash\ P \wedge Q \rightarrow P$ $\qquad$ $\vDash\ P \wedge Q \rightarrow Q$

$\vDash\ P \rightarrow P \vee Q$ $\qquad$ $\vDash\ Q \rightarrow P \vee Q$

$\vDash\ (P \rightarrow Q) \vee (Q \rightarrow R)$

$\vDash\ \neg P \rightarrow (P \rightarrow Q)$ *Conditional addition*

$\vDash\ Q \rightarrow (P \rightarrow Q)$ *Conditional addition*

$P \rightarrow Q$

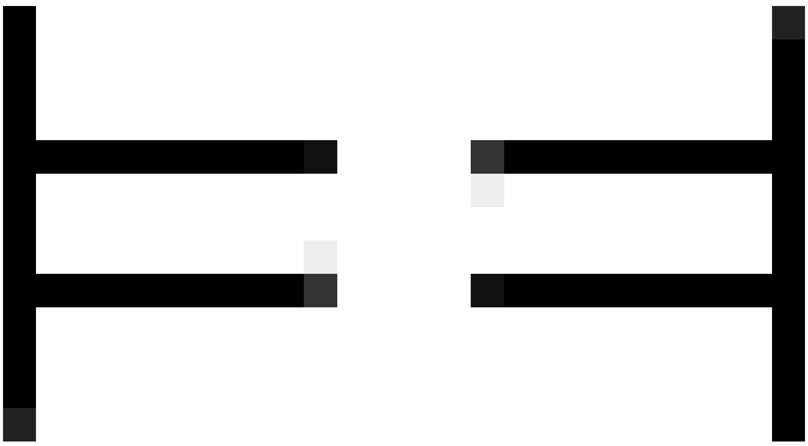

Figure 40 is equivalent to

$\neg Q \rightarrow \neg P$ *Contraposition*

$P \wedge Q \rightarrow R$

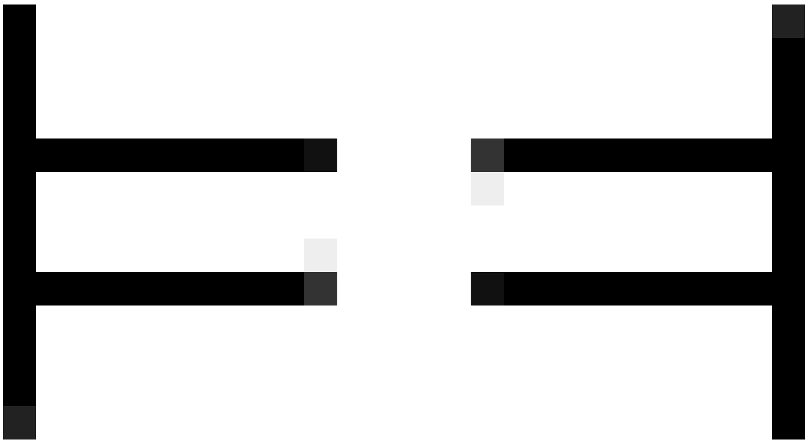

Figure 41 is equivalent to

$P \to (Q \to R)$ *Exportation*

8.3.2 Biconditionals

These tautologies and equivalences are mostly about biconditionals.

$\models\ P \wedge Q \to (P \leftrightarrow Q)$ *Biconditional addition*

$\models\ \neg P \wedge \neg Q \to (P \leftrightarrow Q)$ *Biconditional addition*

$\models (P \leftrightarrow Q) \vee (P \leftrightarrow \neg Q)$

$\neg(P \leftrightarrow Q)$

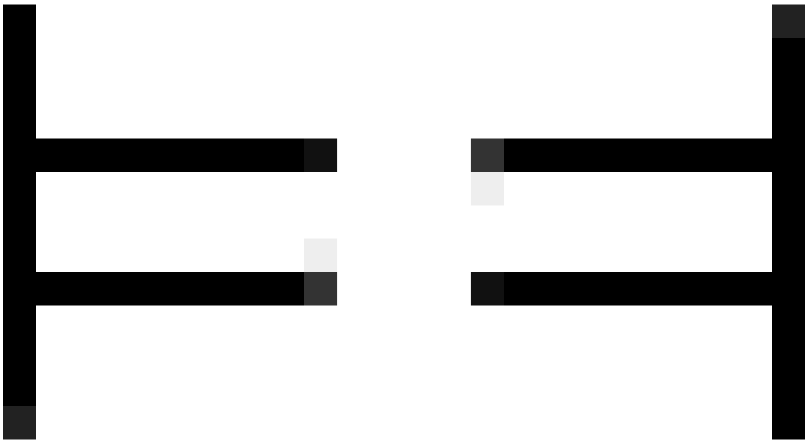

Figure 42 is equivalent to

$\neg P \leftrightarrow Q$

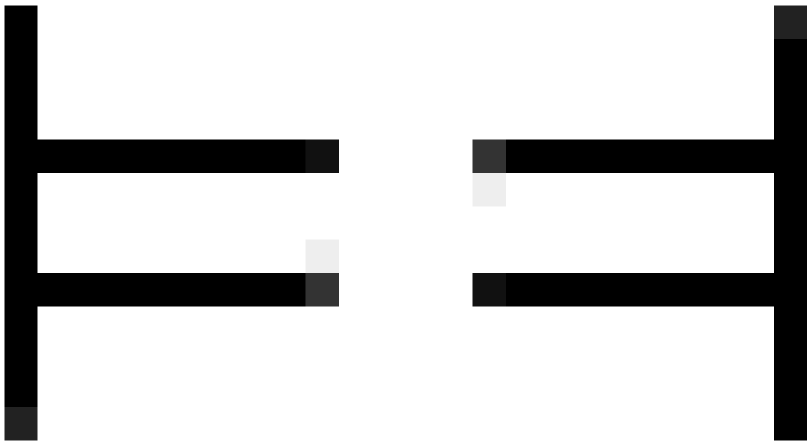

Figure 43 is equivalent to

$P \leftrightarrow \neg Q$

8.3.3 Miscellaneous

We repeat DeMorgan's Laws from the Interdefinability of connectives[2] section of Expressibility[3] and add two additional forms. We also list some additional tautologies and equivalences.

$\vDash\ P \leftrightarrow P \land P$ *Idempotence*[4] *for conjunction*

$\vDash\ P \leftrightarrow P \lor P$ *Idempotence for disjunction*

$\vDash\ P \rightarrow P \lor Q$ *Disjunctive addition*

$\vDash\ Q \rightarrow P \lor Q$ *Disjunctive addition*

$\vDash\ P \land \neg P \rightarrow Q$

$P \land Q$

2 Chapter 7.4 on page 57
3 Chapter 6.6 on page 49
4 https://en.wikipedia.org/wiki/Idempotence

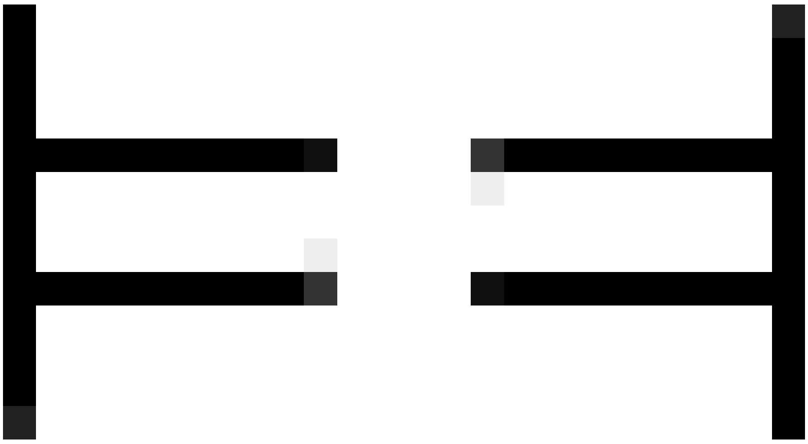

Figure 44

$\neg(\neg P \vee \neg Q)$ *Demorgan's Laws*

$P \vee Q$

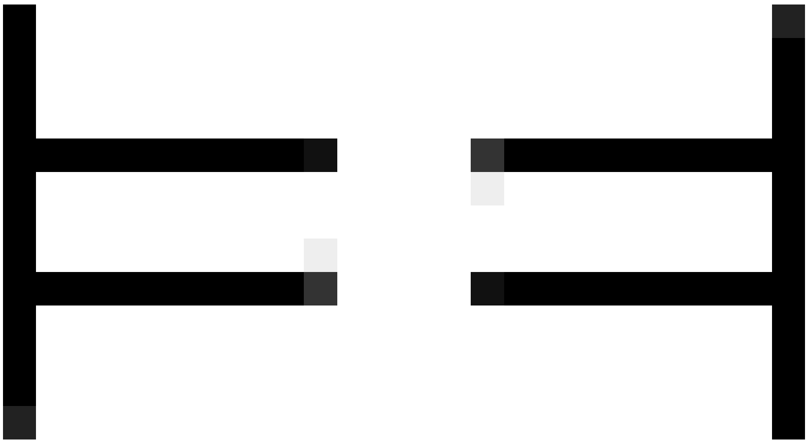

Figure 45

$\neg(\neg P \wedge \neg Q)$ *Demorgan's Laws*

$\neg(P \wedge Q)$

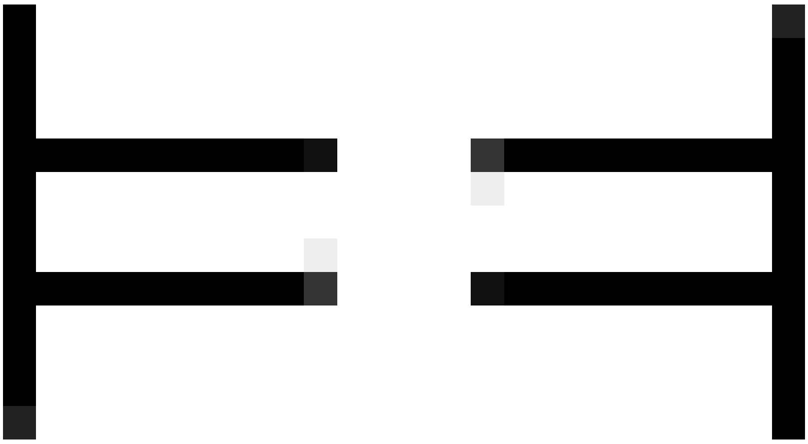

Figure 46

$\neg P \vee \neg Q$ *Demorgan's Laws*

$\neg(P \vee Q)$

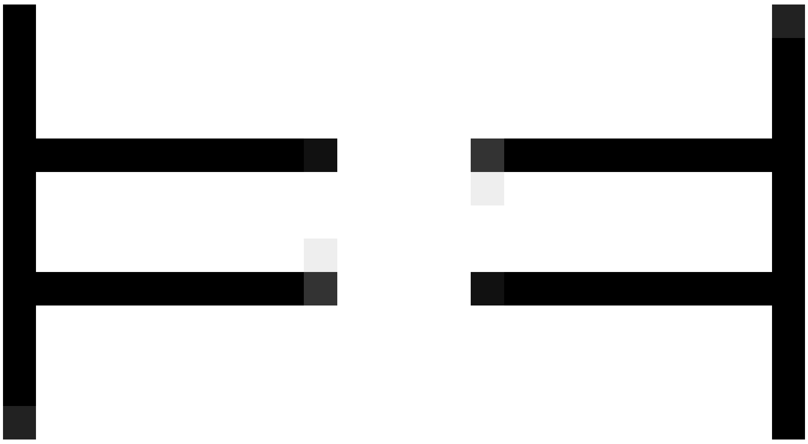

Figure 47

$\neg P \wedge \neg Q$ *Demorgan's Laws*

P

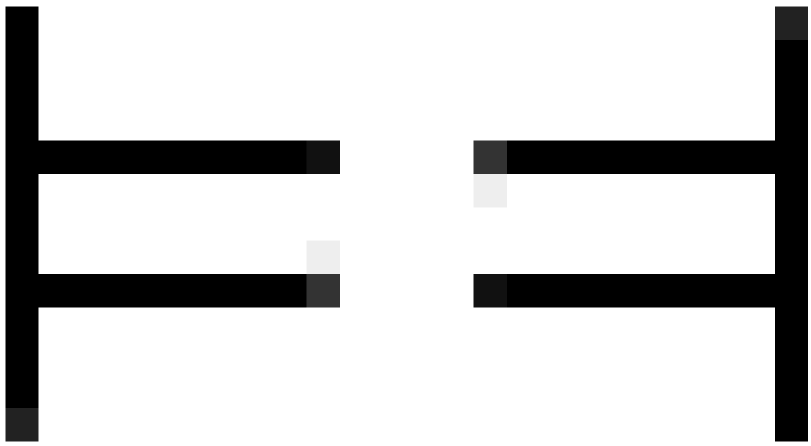

Figure 48 is equivalent to

¬¬P *Double Negation*

8.4 Deduction and reduction principles

The following two principles will be used in constructing our derivation system on a later page. They can easily be proven, but—since they are neither tautologies nor equivalences—it takes more than a mere truth table to do so. We will not attempt the proof here.

8.4.1 Deduction principle

Let φ and ψ both be formulae, and let Γ be a set of formulae.

$$\text{If } \Gamma \cup \{\varphi\} \vDash \psi, \text{ then } \Gamma \vDash (\varphi \rightarrow \psi)$$

8.4.2 Reduction principle

Let φ and ψ both be formulae, and let Γ be a set of formulae.

$$\text{If } \Gamma \cup \{\varphi\} \vDash \psi \text{ and } \Gamma \cup \{\varphi\} \vDash \neg\psi, \text{ then } \Gamma \vDash \neg\varphi,$$

$$\text{If } \Gamma \cup \{\neg\varphi\} \vDash \psi \text{ and } \Gamma \cup \{\neg\varphi\} \vDash \neg\psi, \text{ then } \Gamma \vDash \varphi$$

9 Substitution and Interchange

This page will use the notions of *occurrence* and *subformula* introduced at the Additional terminology[1] section of Formal Syntax[2]. These notions have been little used if at all since then, so you might want to review them.

9.1 Substitution

9.1.1 Tautological forms

We have introduced a number of tautologies, one example being

(1) $\mathrm{Q} \rightarrow (\mathrm{P} \rightarrow \mathrm{Q})$

This has the (informally written) form

(2) $\psi \rightarrow (\varphi \rightarrow \psi)$

As it turns out, any formula matching this form is a tautology. Thus, for example,

(3) $\mathrm{R} \land \mathrm{S} \rightarrow (\mathrm{P} \lor \mathrm{Q} \rightarrow \mathrm{R} \land \mathrm{S})$

is a tautology. This process is general. Take any tautology. Find its most fully explicitly form by uniformly replacing distinct sentence letters with distinct Greek letters. We can call this a *tautological form*, which will not be a formula but rather a metalogical expression. Any instance of this tautological form is a tautology.

9.1.2 Substitution instances

The preceding illustrated how we can generate new tautologies from old ones via tautological forms. Here, we will show how to generate tautologies without resort to tautological forms. To do this, we will define a substitution instance of a formula. Any substitution instance of a tautology is also a tautology.

First, we define the simple substitution instance of a formula for a sentence letter. Let φ and ψ be formulae and π be a sentence letter. The *simple substitution instance* $\varphi[\pi/\psi]$ is the result of replacing *every* occurrence of π in φ with an occurrence of ψ. A *substitution instance of formulae for a sentence letters* is the result of a chain of simple substitution instances. In particular, a chain of zero simple substitutions instances starting from φ is

1 Chapter 3.5 on page 19
2 Chapter 2.4 on page 15

a substitution instance and indeed is just φ itself. Thus, any formula is a substitution instance of itself.

It turns out that if φ is a tautology, then so is any simple substitution instance $\varphi[\pi/\psi]$. If we start with a tautology and generate a chain of simple substitution instances, then every formula in the chain is also a tautology. Thus any (not necessarily simple) substitution instance of a tautology is also a tautology.

9.1.3 Substitution examples

Consider (1) again. We substitute $\mathrm{R} \wedge \mathrm{S}$ for every occurrence of Q in (1). This gives us the following simple substitution instance of (1):

(4) $\mathrm{R} \wedge \mathrm{S} \to (\mathrm{P} \to \mathrm{R} \wedge \mathrm{S})$

In this, we substitute $\mathrm{P} \vee \mathrm{Q}$ for P. That gives us (3) as a simple substitution instance of (4). Since (3) is the result of a chain of two simple substitution instances, it is a (non-simple) substitution instance of (1) Since (1) is a tautology, so is (3). We can express the chain of substitutions as

$$\mathrm{Q} \to (\mathrm{P} \to \mathrm{Q})[\mathrm{Q}/\mathrm{R} \wedge \mathrm{S}][\mathrm{P}/\mathrm{P} \vee \mathrm{Q}]$$

Take another example, also starting from (1). We want to obtain

(5) $\mathrm{P} \to (\mathrm{Q} \to \mathrm{P})$

Our first attempt won't work. First we substitute Q for P obtaining

$$\mathrm{Q} \to (\mathrm{Q} \to \mathrm{Q})$$

Next we substitute P for Q obtaining

(6) $\mathrm{P} \to (\mathrm{P} \to \mathrm{P})$

This is indeed a tautology, but it is not the one we wanted. Let's try again. In (1), we substitute R for P obtaining

$$\mathrm{Q} \to (\mathrm{R} \to \mathrm{Q})$$

Now substitute P for Q obtaining

$$\mathrm{P} \to (\mathrm{R} \to \mathrm{P})$$

Finally, substituting Q for R gets us the result we wanted, namely (5). Since (1) is a tautology, so is (5). We can express the chain of substitutions as

$$\mathrm{Q} \to (\mathrm{P} \to \mathrm{Q})[\mathrm{P}/\mathrm{R}][\mathrm{Q}/\mathrm{P}][\mathrm{R}/\mathrm{Q}]$$

9.1.4 Simultaneous substitutions

We can compress a chain of simple substitutions into a single complex substitution. Let φ, ψ_1, ψ_2, ... be formulae; let π_1, π_2, ... be sentence letters. We define a *simultaneous substitution instance of formulas for sentence letters* $\varphi[\pi_1/\psi_1,\ \pi_2/\psi_2,\ ...]$ be the result of starting with φ and simultaneously replacing π_1 with ψ_1, π_2 with ψ_2, ... We can regenerate our examples.

The previously generated formula (3) is

$$\mathrm{Q} \rightarrow (\mathrm{P} \rightarrow \mathrm{Q})[\mathrm{P}/\mathrm{P} \vee \mathrm{Q}, \mathrm{Q}/\mathrm{R} \wedge \mathrm{S}]$$

Similarly, (5) is

$$\mathrm{Q} \rightarrow (\mathrm{P} \rightarrow \mathrm{Q})[\mathrm{P}/\mathrm{Q},\ \mathrm{Q}/\mathrm{P}]$$

Finally (6) is

$$\mathrm{Q} \rightarrow (\mathrm{P} \rightarrow \mathrm{Q})[\mathrm{Q}/\mathrm{P}]$$

When we get to predicate logic, simultaneous substitution instances will not be available. That is why we defined *substitution instance* by reference to a chain of simple substitution instances rather than as a simultaneous substitution instance.

9.2 Interchange

9.2.1 Interchange of equivalent subformulae

We previously saw the following equivalence at Properties of Sentential Connectives[3]:

(7) P

3 Chapter 8.3.3 on page 84

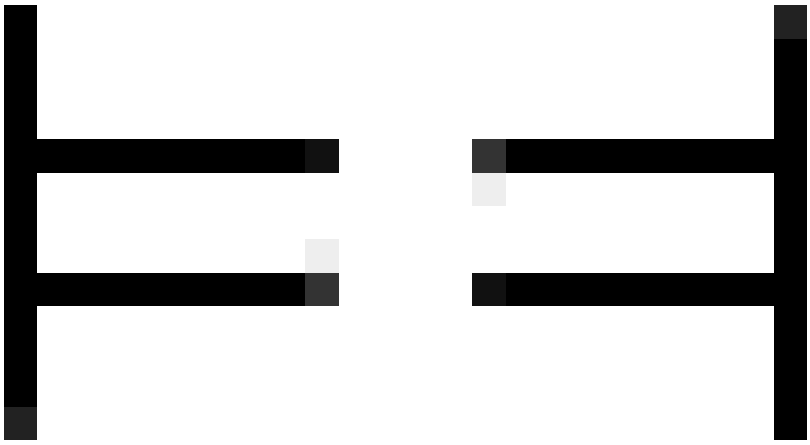

Figure 49 is equivalent to

$\neg\neg P$

You then might expect the following equivalence:

$P \rightarrow Q \wedge R$

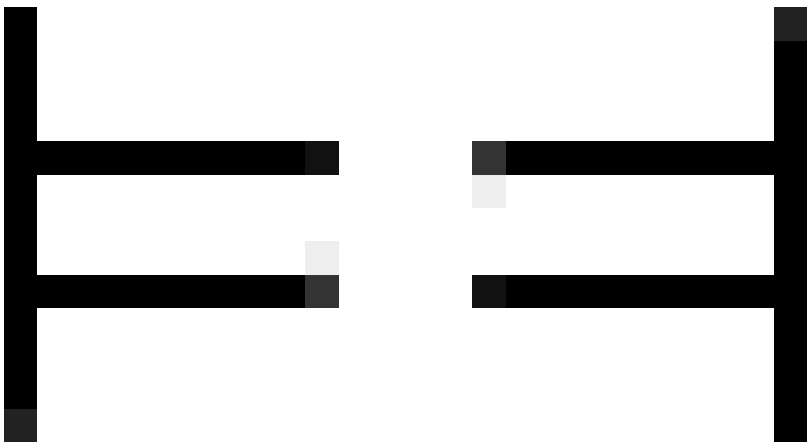

Figure 50 is equivalent to

$\neg\neg(P \rightarrow Q \wedge R)$

This expectation is correct; the two formulae are equivalent. Let φ and ψ be equivalent formulae. Let χ_1 be a formula in which φ occurs as a subformula. Finally, let χ_2 be the result of replacing in χ at least one (not necessarily all) occurrences of φ with ψ. Then χ_1 and χ_2 are equivalent. This replacement is called an *interchange*.

For a second example, suppose we want to generate the equivalence

(8) $\quad P \rightarrow Q \wedge R$

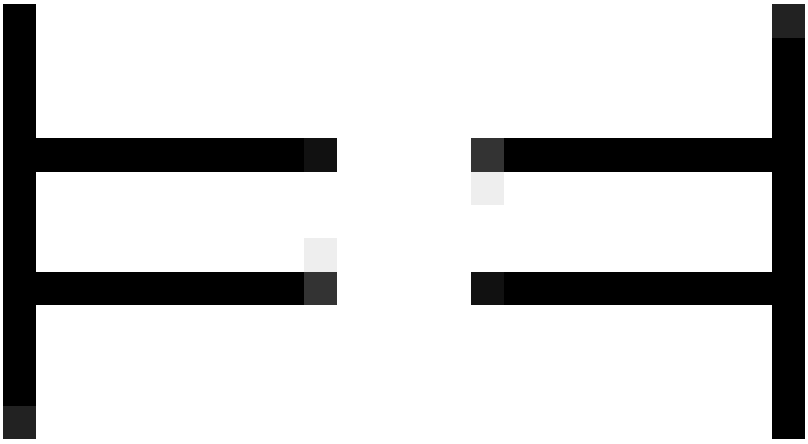

Figure 51 is equivalent to

$P \rightarrow \neg\neg(Q \wedge R)$

We note the following equivalence:

(9) $\quad Q \wedge R$

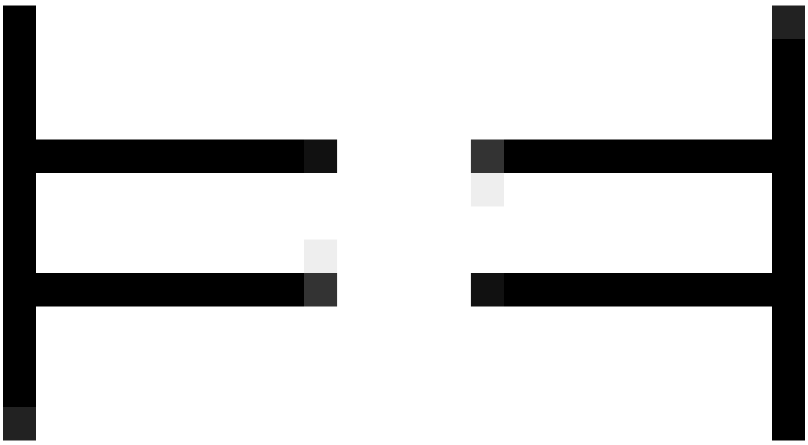

Figure 52 is equivalent to

$\neg\neg(Q \wedge R)$

These two formulae can be confirmed to be equivalent either by truth table or, more easily, by substituting $\mathrm{Q} \land \mathrm{R}$ for P in both formulae of (7).

This substitution does indeed establish (9) as an equivalence. We already noted that φ and ψ are equivalent if and only if $\varphi \leftrightarrow \psi$ is a tautology. Based on (7), we get the tautology

$$\mathrm{P} \leftrightarrow \neg\neg\mathrm{P}$$

Our substitution then yields

$$\mathrm{Q} \land \mathrm{R} \leftrightarrow \neg\neg(\mathrm{Q} \land \mathrm{R})$$

which is also a tautology. The corresponding equivalence is then (9).

Based on (9), we can now replace the consequent of $\mathrm{P} \rightarrow \mathrm{Q} \land \mathrm{R}$ with its equivalent. This generates the desired equivalence, namely (8).

Every formula equivalent to a tautology is also a tautology. Thus an interchange of equivalent subformulae within a tautology results in a tautology. For example, we can use the substitution instance of (7):

Q

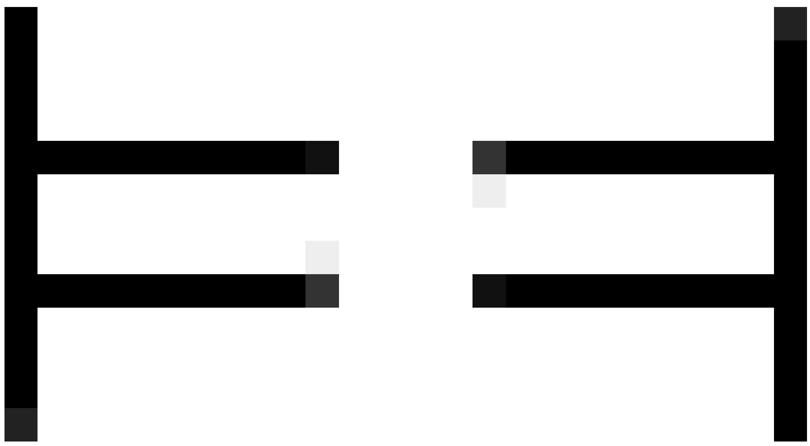

Figure 53 is equivalent to

$\neg\neg$Q

together with the tautology previously seen at Properties of Sentential Connectives[4]:

$$(\mathrm{P} \rightarrow \mathrm{Q}) \lor (\mathrm{Q} \rightarrow \mathrm{R})$$

4 Chapter 8.3.1 on page 82

to obtain

$$(P \rightarrow \neg\neg Q) \vee (Q \rightarrow R)$$

as a new tautology.

9.2.2 Interchange example

As an example, we will use the interdefinability of connectives[5] to express

(10) $P \leftrightarrow Q \vee R$

using only conditionals and negations.

Based on

$P \vee Q$

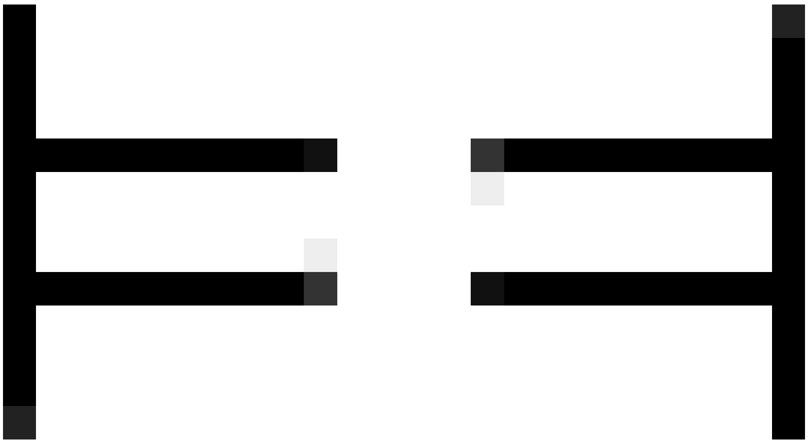

Figure 54 is equivalent to

$\neg P \rightarrow Q$

we get the substitution instance

$Q \vee R$

5 Chapter 7.4 on page 57

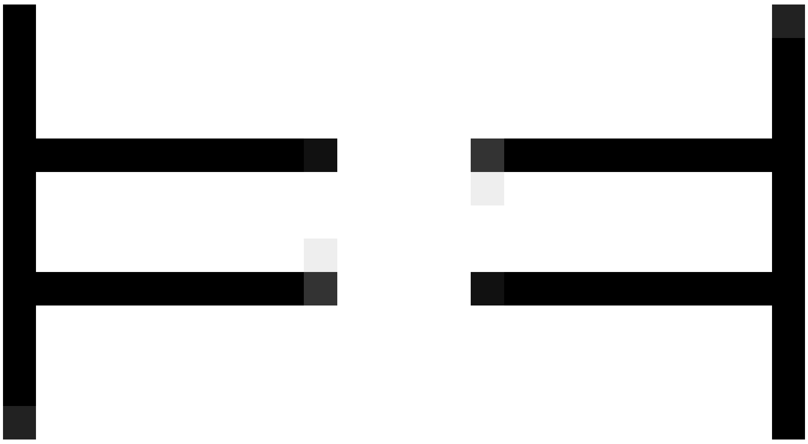

Figure 55 is equivalent to

$\neg Q \to R$

which in turn allows us to replace the appropriate subformula in (10) to get:

(11) $P \leftrightarrow \neg Q \to R$

The equivalence

$P \leftrightarrow Q$

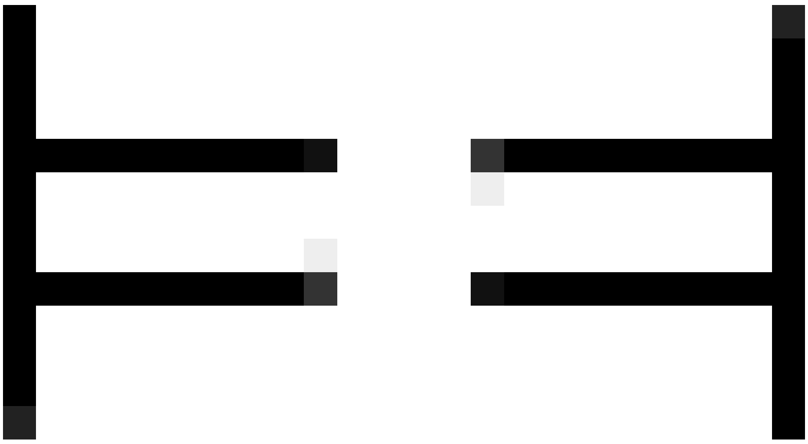

Figure 56 is equivalent to

$(P \to Q) \wedge (Q \to P)$

together with the appropriate substitution gives us

(12) $(P \rightarrow (\neg Q \rightarrow R)) \wedge ((\neg Q \rightarrow R) \rightarrow P)$

as equivalent to (11).

Finally, applying

$P \wedge Q$

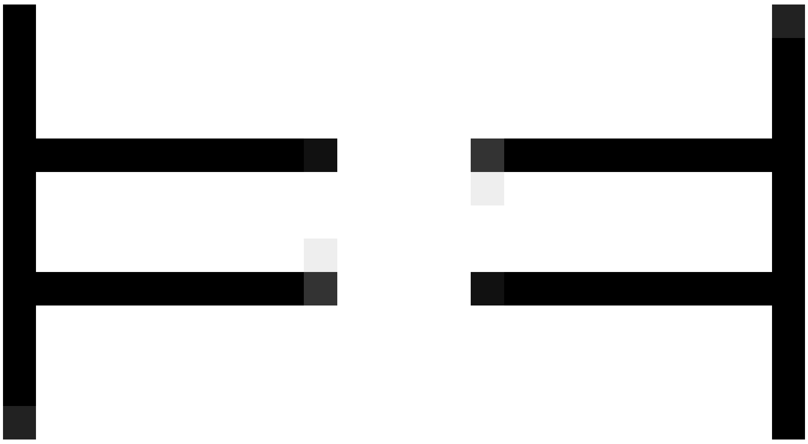

Figure 57

$\neg(P \rightarrow \neg Q)$

together with the appropriate substitution, yields our final result:

$$\neg(P \rightarrow (\neg Q \rightarrow R) \rightarrow \neg((\neg Q \rightarrow R) \rightarrow P))$$

9.3 Summary

This page has presented two claims.

- A substitution instance of a tautology is also a tautology.
- Given a formula, the result of interchanging a subformula with an equivalent is a formula equivalent to the given formula.

These claims are not trivial observations or the result of a simple truth table. They are substantial claims that need proof. Proofs are available in a number of standard metalogic textbooks, but are not presented here.

10 Translations

The page The Sentential Language[1] gave a very brief look at translation between English and $\mathcal{L_S}$. We look at this in more detail here.

10.1 English sentential connectives

In the following discussion, we will assume the following assignment of English sentences to $\mathcal{L_S}$ sentence letters:

P : 2 is a prime number.

Q : 2 is an even number.

R : 3 is an even number.

10.1.1 Not

The canonical translation of $\neg$ into English is 'it is not the case that'. Given the assignment above,

(1) $\neg$P

translates as

It is not the case that 2 is a prime number.

But we usually express negation in English simply by 'not' or by adding the contraction 'n't' to the end of a word. Thus (1) can also translate either of:

2 is not a prime number.

2 isn't a prime number.

10.1.2 If

The canonical translation of $\rightarrow$ into English is 'if ... then ...'. Thus

(2) P $\rightarrow$ Q

translates into English as

(3) If 2 is a prime number, then 2 is an even number.

1 Chapter 1.3.3 on page 9

Objections have been raised to the canonical translation, and our example may illustrate the problem. It may seem odd to count (3) as true; however, our semantic rules does indeed count (2) as true (because both P and Q are true). We might expect that, if a conditional and its antecedent are true, the consequent is true *because* the antecedent is. Perhaps we expect a general rule

(4) if x is a prime number, then x is an even number

to be true—but this rule is clearly false. In any case, we often expect the truth of the antecedent (if it is indeed true) to be somehow *relevant* to the truth of the conclusion (if that is indeed true). (2) is an exception to the usual relevance of a number being prime to a number being even.

The $\rightarrow$ conditional of $\mathcal{L_S}$ is called the *material conditional* in contrast to strict conditional[2] or counterfactual conditional[3]. Relevance logic[4] attempts to define a conditional which meets these objections. See also the Stanford Encyclopedia of Philosophy entry on relevance logic[5].

It is generally accepted today that not all aspects of an expression's linguistic use are part of its linguistic meaning. Some have suggested that the objections to reading 'if' as a material conditional are based on *conversational implicature* and so not based on the meaning of 'if'. See the Stanford Encyclopedia of Philosophy entry on implicature[6] for more information. As much as a simplifying assumption than anything else, we will adopt this point of view. We can also point out in our defense that translations need not be exact to be useful. Even if our simplifying assumption is incorrect, $\rightarrow$ is still the closest expression we have in $\mathcal{L_S}$ to 'if'. It should also be noted that, in mathematical statements and proofs, mathematicians *always* use 'if' as a material conditional. They accept (2) and (3) as translations of each other and do not find it odd to count (3) as true.

'If' can occur at the beginning of the conditional or in the middle. The 'then' can be missing. Thus both of the following (in addition to (3)) translate as (2).

If 2 is a prime number, 2 is an even number.

2 is an even number if 2 is a prime number.

10.1.3 Implies

We do not translate 'implies' into $\mathcal{L_S}$. In particular, we reject

2 is a prime number implies 2 is an even number.

as grammatically ill-formed and therefore not translatable as (2). See the Implication[7] section of Validity[8] for more details.

2 https://en.wikipedia.org/wiki/Strict%20conditional
3 https://en.wikipedia.org/wiki/Counterfactual%20conditional
4 https://en.wikipedia.org/wiki/Relevance%20logic
5 http://plato.stanford.edu/entries/logic-relevance/
6 http://plato.stanford.edu/entries/implicature/
7 Chapter 6.6 on page 49
8 Chapter 6.2 on page 45

10.1.4 Only if

The English

(5) 2 is a prime number only if 2 is an even number

is equivalent to the English

If 2 is not an even number, then 2 is not a prime number.

This, in turn, translates into $\mathcal{L}_\mathcal{S}$ as

(6) $\neg Q \rightarrow \neg P$

We saw at Conditionals[9] section of Properties of Sentential Connectives[10] that (6) is equivalent to

(7) $P \rightarrow Q$

Many logic books give this as the preferred translation of (5) into $\mathcal{L}_\mathcal{S}$. This allows the convenient rule ''if' always introduces an antecedent while 'only if' always introduces a consequent'.

Like 'if', 'only if' can appear in either the first or middle position of a conditional. (5) is equivalent to

Only if 2 is an even number, is 2 a prime number.

10.1.5 Provided that

'Provided that'—and similar expressions such as 'given that' and 'assuming that'—can be use equivalently with 'if'. Thus each of the following translate into $\mathcal{L}_\mathcal{S}$ as (2).

2 is an even number provided that 2 is a prime number.

2 is an even number assuming that 2 is a prime number.

Provided that 2 is a prime number, 2 is an even number.

Prefixing 'provided that' with 'only' works the same as prefixing 'if' with only. Thus each of the following translate into $\mathcal{L}_\mathcal{S}$ as (6) or (7).

2 is a prime number only provided that 2 is an even number.

2 is a prime number only assuming that 2 is an even number.

Only provided that 2 is an even number, is 2 a prime number

10.1.6 Or

The canonical translation of $\vee$ into English is '[either] ... or ...' (where the 'either' is optional). Thus

9 Chapter 8.3.1 on page 82
10 Chapter 7.5.2 on page 71

(8) $P \vee Q$

translates into English as

(9) 2 is a prime number or 2 is an even number

or

Either 2 is a prime number or 2 is an even number.

We saw at the Interdefinability of connectives[11] section of Expressibility[12] that (8) is equivalent to

$$\neg P \rightarrow Q$$

Just as there were objections to understanding 'if' as $\rightarrow$, there are similar objections to understanding 'or' as $\vee$. We will again make the simplifying assumption that we can ignore these objections.

The English 'or' has both an inclusive and—perhaps somewhat more controversially—an exclusive use. The *inclusive or* is true when at least one disjunct is true; the *exclusive or* is true when exactly one disjunct is true. Our $\vee$ matches the inclusive use. The inclusive use becomes especially apparent in negations. If President Bush promises not to invade Iran or North Korea, not even the best Republican spin doctors will claim he can keep his promise by invading both. The exclusive reading of (9) translates into $\mathcal{L}_\mathcal{S}$ as

$$(P \vee Q) \wedge \neg(P \wedge Q)$$

or more simply (and less intuitively) as

$$P \leftrightarrow \neg Q$$

In English, telescoping is possible with 'or'. Thus, (8) translates

2 is either a prime number or an even number.

Similarly,

$$Q \vee R$$

translates

2 or 3 is an even number.

11 Chapter 7.4 on page 57
12 Chapter 6.6 on page 49

10.1.7 Unless

'Unless' has the same meaning as 'if not'. Thus

(10) $\neg Q \rightarrow P$

translates

(11) 2 is a prime number unless 2 is an even number

and

(12) Unless 2 is an even number, 2 is a prime number.

We saw at the Interdefinability of connectives[13] section of Expressibility[14] that (10) is equivalent to (8). Many logic books give (8) as the preferred translation of (11) or (12) into $\mathcal{L}_\mathcal{S}$.

10.1.8 Nor

At the Joint denial[15] section of Expressibility[16], we temporarily added $\downarrow$ to $\mathcal{L}_\mathcal{S}$ as the connective for joint denial. If we had that connective still available to us, we could translate

Neither 2 is a prime number nor 2 is an even number

as

$$P \downarrow Q$$

However, since $\downarrow$ is not really in the vocabulary of $\mathcal{L}_\mathcal{S}$, we need to paraphrase. Either of the following will do:

(13) $\neg(P \vee Q)$.

(14) $(\neg P \wedge \neg Q)$.

The same telescoping applies as with 'or'.

2 is neither a prime number nor an even number

translates into $\mathcal{L}_\mathcal{S}$ as either (13) or (14). Similarly,

Neither 2 nor 3 is an even number

translates as either of

$$\neg(Q \vee R)$$

13 Chapter 7.4 on page 57
14 Chapter 6.6 on page 49
15 Chapter 7.5.2 on page 67
16 Chapter 6.6 on page 49

$$(\neg Q \wedge \neg R)$$

10.1.9 And

The canonical translation of $\wedge$ into English is '[both] ... and ...' (where the 'both' is optional'). Thus

(15) $P \wedge Q$

translates into English as

2 is a prime number and 2 is an even number

or

Both 2 is a prime number and 2 is an even number.

Our translation of 'and' as $\wedge$ is not particularly controversial. However, 'and' is sometimes used to convey temporal order. The two sentences

She got married and got pregnant.

She got pregnant and got married.

are generally heard rather differently.

'And' has the same telescoping as 'or'.

2 is a both prime number and an even number

translates into $\mathcal{L}_{\mathcal{S}}$ as (15)

Both 2 and 3 are even numbers

translates as

$$Q \wedge R$$

10.1.10 If and only if

The canonical translation of $\leftrightarrow$ into English is '... if and only if ...'. Thus

(16) $P \leftrightarrow Q$

translates into English as

2 is a prime number if and only if 2 is an even number.

The English

(17) 2 is a prime number if and only if 2 is an even number

is a shortened form of

2 is a prime number if 2 is an even number, and 2 is a prime number only if 2 is an even number

which translates as

$$(Q \rightarrow P) \land (\neg Q \rightarrow \neg P))$$

or more perspicaciously as the equivalent formula

(18) $(P \rightarrow Q) \land (Q \rightarrow P))$.

We saw at the Interdefinability of connectives[17] section of Expressibility[18] that (18) is equivalent to (16). Issues concerning the material versus non-material interpretations of 'if' apply to 'if and only if' as well.

10.1.11 Iff

Mathematicians and sometimes others use 'iff' as an abbreviated form of 'if and only if'. So

2 is a prime number iff 2 is an even number

abbreviates (17) and translates as (16).

10.2 Examples

17 Chapter 7.4 on page 57
18 Chapter 6.6 on page 49

11 Derivations

11.1 Derivations

At Validity[1], we introduced the notion of *validity* for formulae and for arguments. In sentential logic, a valid formula is a tautology.

Up to now, we could show a formula φ to be valid (a tautology) in the following ways.

- Do a truth table for φ.
- Obtain φ as a substitution instance of a formula already known to be valid.
- Obtain φ by applying interchange of equivalents to a formula already know to be valid.

Truth tables become unavailable in predicate logic. Without an alternate method, there will be no way of getting the second two methods started since they need already known validities to work. Derivations provide such an alternate method for showing a formula valid, a method that continues to work even after truth tables become unavailable. This page, together with the several following it, introduce this technique. Note, the claim that a derivation shows an argument valid assumes a sound derivation system, see *soundness* below.

A derivation is a series of numbered lines, each line consisting of a formula with an annotation. The annotations provide the justification for adding the line to the derivation. A derivation is a highly formalized analogue to—or perhaps a model of—a mathematical proof.

A typical derivation system will allow some of the following types of lines:

- A line may be an axiom. The derivation system may specify a set of formula as axioms. These are accepted as true for any derivation. For sentential logic the set of axioms is a fixed subset of tautologies.
- A line may be an assumption. A derivation may have several types of assumptions. The following cover the standard cases.
  - A premise. When attempting to show the validity of an argument, a premise of that argument may be assumed.
  - A temporary assumption for use in a subderivation. Such assumptions are intended to be active for only part of a derivation and must be discharged (made inactive) before the derivation is considered complete. Subderivations will be introduced on a later page.
- A line may result from applying an inference rule to previous lines. An *inference* is a syntactic transformation of previous lines to generate a new line. Inferences are required

1 Chapter 6.2 on page 45

to follow one of a fixed set of patterns defined by the derivation system. These patterns are the system's *inference rules*. The idea is that any inference fitting an inference rule should be a valid argument.

11.2 Soundness and validity

We noted at Formal Semantics[2] that a formal language such as $\mathcal{L}_{\mathcal{S}}$ can be interpreted via several alternative and even competing semantic rule-sets. Multiple derivation systems can be also defined for a given syntax-semantics pair. A triple consisting of a formal syntax, a formal semantics, and a derivation system is a *logical system*.

A derivation is intended to show an argument to be valid. A derivation of a zero-premise argument is intended to show its conclusion to be a valid formula—in sentential logic this means showing it to be a tautology. These are the intentions. Given a logical system, the derivation system is *sound* if and only if it achieves this goal. That is, a derivation system is *sound* (has the property of *soundness*) if and only if every formula (and argument) derivable in its derivation system is valid (given a syntax and a semantics).

Another desirable quality of a derivation system is completeness. Given a logical system, its derivation system is *complete* if and only if every valid formula is derivable. However, there are some logics for which no derivation system is or can be complete.

Soundness and completeness are metalogic substantial results. Their proofs will not be given here, but are available in many standard metalogic text books.

11.3 Turnstiles

The $\vDash$ symbol is sometimes called a *turnstile*, in particular a *semantic turnstile*. We previously introduced the following three uses of this symbol.

(1)	$\mathfrak{v} \vDash \varphi$	$\mathfrak{v}$ satisfies φ.
(2)	$\vDash \varphi$	φ is valid.
(3)	$\Gamma \vDash \varphi$	Γ implies (has as a logical consequence) φ.

Derivations have a counterpart to the semantic turnstile, namely the *syntactic turnstile*. (1) above has no syntactic counterpart. However, (2) and (3) above have the following counterparts.

(4)	$\vdash \varphi$	φ is provable.
(5)	$\Gamma \vdash \varphi$	Γ proves (has as a derivational consequence) φ.

(4) is the case if and only if there is a correct derivation of φ from no premises. Similarly, (5) is the case if and only if there is a correct derivation of φ which takes the members of Γ as premises.

The negations of (4) and (5) above are

(6) $\nvdash \varphi$

2 Chapter 4.3 on page 24

(7) $$\Gamma \nvdash \varphi$$

We can now define soundness and completeness as follows:

- Given a logical system, its derivation system is *sound* if and only if:

$$\text{If } \Gamma \vdash \varphi, \text{ then } \Gamma \vDash \varphi.$$

- Given a logical system, its derivation system is *complete* if and only if:

$$\text{If } \Gamma \vDash \varphi, \text{ then } \Gamma \vdash \varphi.$$

12 Inference Rules

12.1 Overview

Inference rules will be formated as in the following example.

Conditional Elimination (CE)

$$(\varphi \to \psi)$$

$$\underline{\varphi \qquad}$$

$$\psi$$

The name of this inference rule is 'Conditional Elimination', which can be abbreviated as 'CE'. We can apply this rule if formulae having the forms above the line appear as active lines in the derivation. These are called the *antecedent lines* for this inference. Applying the rule adds a formula having the form below the line. This is called the *consequent line* for this inference. The annotation for the newly derived line will be the line numbers of the antecedent lines and the abbreviation 'CE'.

Note. You might see *premise line* and *conclusion line* for *antecedent line* and *consequent line*. You may see other terminology as well. Most textbooks avoid giving any special terminology here; they just leave it up to the classroom teaching assistant to make it up as they go.

Each sentential connective will have two inference rules, one each of the following types.

- An *introduction rule*. The introduction rule for a given connective allows us to derive a formula having the given connective as its main connective.
- An *elimination rule*. The elimination rule for a given connective allows us to use a formula already appearing in the derivation having the given connective as its main connective.

Three rules (Negation Introduction, Negation Elimination, and Conditional Introduction) will be deferred to a later page. These are so-called discharge rules which will be explained when we get to subderivations.

Three rules (Conjunction Elimination, Disjunction Introduction, and Biconditional Elimination) will have two forms each. We somewhat arbitrarily count the two patterns as forms of the same rule rather than separate rules.

The validity of the inferences on this page can be shown by truth table.

12.2 Inference rules

12.2.1 Negation

Negation Introduction (NI)

Deferred to a later page.

Negation Elimination (NE)

Deferred to a later page.

12.2.2 Conjunction

Conjunction Introduction (KI)

$$\frac{\begin{array}{c}\varphi \\ \psi\end{array}}{(\varphi \wedge \psi)}$$

Conjunction Introduction traditionally goes by the name *Adjunction* or *Conjunction*.

Conjunction Elimination, Form I (KE)

$$\frac{(\varphi \wedge \psi)}{\varphi}$$

Conjunction Elimination, Form II (KE)

$$\frac{(\varphi \wedge \psi)}{\psi}$$

Conjunction Elimination traditionally goes by the name *Simplification*.

12.2.3 Disjunction

Disjunction Introduction, Form I (DI)

$$\frac{\varphi}{(\varphi \vee \psi)}$$

Disjunction Introduction, Form II (DI)

$$\frac{\psi}{(\varphi \vee \psi)}$$

Disjunction Introduction traditionally goes by the name *Addition*.

Disjunction Elimination (DE)

$$\varphi \vee \psi$$

$$(\varphi \to \chi)$$

$$\frac{(\psi \to \chi)}{\chi}$$

Disjunction Elimination traditionally goes by the name *Separation of Cases*.

12.2.4 Conditional

Conditional Introduction (CI)

Deferred to a later page.

Conditional Elimination (CE)

$$(\varphi \to \psi)$$

$$\frac{\varphi}{}$$

$$\psi$$

Conditional Elimination traditionally goes by the Latin name *Modus Ponens* or, less often, by *Affirming the Antecedent*.

12.2.5 Biconditional

Biconditional Introduction (BI)

$$(\varphi \to \psi)$$

$$\underline{(\psi \to \varphi)}$$

$$(\varphi \leftrightarrow \psi)$$

Biconditional Elimination, Form I (BE)

$$(\varphi \leftrightarrow \psi)$$

$$\underline{\varphi \qquad\quad}$$

$$\psi$$

Biconditional Elimination, Form II (BE)

$$(\varphi \leftrightarrow \psi)$$

$$\underline{\psi \qquad\quad}$$

$$\varphi$$

12.3 Examples

Inference rules are easy enough to apply. From the lines

(1) $P \wedge Q \to (S \leftrightarrow T)$

and

(2) $P \wedge Q$

we can add to a derivation

(3) $S \leftrightarrow T$.

The annotation will be the line numbers of (1) and (2) and the abbreviation for Conditional Elimination, namely 'CE'. The order of the antecedent lines does not matter. The inference is allowed if (1) appears before (2); it is also allowed if (2) appears before (1).

It must be remembered that inference rules are strictly syntactical. Semantically obvious variations is not allowed. It is not allowed, for example, to derive (3) from (1) and

(4) $Q \wedge P$

However, you can get from (1) and (4) to (3) by first deriving

(5) Q

and

(6) P

by Conjunction Elimination (KE). Then you can derive (2) by Conjunction Introduction (KI) and finally (3) from (1) and (2) by Conditional Elimination (CE) as before. Some derivation systems have a rule, often called Tautological Implication, allowing you to derive any tautological consequence of previous lines. However, this should be seen as an (admittedly useful) abbreviation. On later pages, we will implement a restricted version of this abbreviation.

It is generally useful to apply break down premises, other assumptions (to be introduced on a later page) by applying elimination rules—and then continue breaking down the results. Supposing that is why we applied CE to (1) and (2), it will likely be useful to derive

(7) $S \to T$

and

(8) $T \to S$

by applying Biconditional Elimination (BE) to (3). To further break this down, you might then attempt to derive S or T so that you can apply CE to (7) or (8).

If you know what line you want to derive, you can build it up by applying introduction rules. That was the strategy for deriving (2) from (5) and (6).

13 Constructing a Simple Derivation

Our derivations consists two types of elements.

- *Derived lines.* A derived line has three parts:
  - *Line number.* This allows the line to be referred to later.
  - *Formula.* The purpose of a derivation is to derive formulae, and this is the formula that has been derived at this line.
  - *Annotation.* This specifies the justification for entering the formula into the derivation.
- *Fencing.* These include:
  - Vertical lines between the line number and the formula. These are used to set off subderivations which we will get to in the next module.
  - Horizontal lines separating premises and temporary assumptions from other lines. When we get to predicate logic, there are restrictions on using premises and temporary assumptions. Setting them off in an easy-to-recognize fashion aids in adhering to the restrictions.

We sometimes are a bit sloppy and speak of the formula as if it were the entire line. But the line also includes the formula's entourage, the line number and the annotation.

13.1 Rules

13.1.1 Premises

The annotation for a premise is 'Premise'. We will require that all premises to be used in the derivation be the first lines. No non-premise line is allowed to appear before a premise. In theory, an argument can have infinitely many premises. However, derivations have only finitely many lines, so only finitely many premises can be used in the derivation. We do not require that *all* premises appear before other lines. This would be impossible for arguments with infinitely many premises. But we do require that all premises to appear in the derivation appear before any other line.

The requirement that premises used in the derivation appear as its first lines is stricter than absolutely necessary. However, certain restrictions that will be needed when we get to predicate logic make the requirement at least a useful convention.

13.1.2 Inference rules

We introduced all but two inference rules in the previous module, and will introduce the other two in the next module.

13.1.3 Axioms

This derivation system does not have any axioms.

13.2 An example derivation

We will construct a derivation for the following argument:

$P \wedge Q,\ P \vee R \rightarrow S,\ S \wedge Q \rightarrow T \quad \therefore \quad T$

First, we enter the premises into the derivation:

1.	$P \wedge Q$	Premise
2.	$P \vee R \to S$	Premise
3.	$S \wedge Q \to T$	Premise

Note the vertical line between the line numbers and the formulae. That is part of the fencing that controls subderivations. We will get to subderivations in the next module. Until then, we simply put a single vertical line the length of the derivation. Note also the horizontal line under the premises. This is fencing that helps distinguish the premises from the other lines in the derivation.

Now we need to use the premises. Applying KE to the first premise twice. we add the following lines:

4.	P	1 KE
5.	Q	1 KE

Now we need to use the second premise by applying CE. Since CE has two antecedent lines, we first need to derive the other line that we will need. We thus add these lines:

6.	$P \vee R$	4 DI
7.	S	2, 6 CE

Now we will use the third premise by applying CE. Again, we first need to derive the other line we will need. The new lines are:

8.	$S \wedge Q$	5, 7 KI
9.	T	3, 8 CE

Note line 9 is T. This is the conclusion of our argument, so we are done. The conclusion does not always fall into our lap so nicely, but here it did. The complete derivation runs:

1.	$P \wedge Q$	Premise
2.	$P \vee R \to S$	Premise
3.	$S \wedge Q \to T$	Premise
4.	P	1 KE
5.	Q	1 KE
6.	$P \vee R$	4 DI
7.	S	2, 6 CE
8.	$S \wedge Q$	5, 7 KI
9.	T	3, 8 CE

14 Subderivations and Discharge Rules

As already seen, we need three more inference rules, Conditional Introduction (CI), Negation Introduction (NI), and Negation Elimination (NE). These require subderivations.

14.1 Deriving conditionals

14.1.1 Example derivation

We begin with an example derivation which illustrates Conditional Introduction. We will provide a derivation for the argument

$$(\mathrm{P} \to \mathrm{Q}) \to \mathrm{R},\ \mathrm{S} \wedge \mathrm{Q} \quad \therefore \quad \mathrm{R}$$

1.	$(P \to Q) \to R$	Premise
2.	$S \wedge Q$	Premise
3.	P	
4.	Q	

Lines 3 and 4 constitute a subderivation. It starts by assuming desired formula's antecedent and ends by deriving the desired formula's consequent. There are two vertical fences between the line numbers and the formulae to set it off from the rest of the derivation and to indicate its subordinate status. Line 3 has a horizontal fence under it to separate the assumption from the rest of the subderivation. Line 5 is the application of Conditional Introduction. It follows not from one or two lines but from the subderivation (lines 3−4) as a whole.

Conditional Introduction is a *discharge rule*. It discharges (makes inactive) that assumption and indeed makes the entire subderivation inactive. Once we apply a discharge rule, no line from the subderivaiton (here, lines 3 and 4) can be further used in the derivation.

14.1.2 The Conditional Introduction rule

Conditional Introduction (CI)

$$\frac{\textit{Assume } \varphi, \textit{ derive } \psi}{(\varphi \rightarrow \psi)}$$

Here, the consequent line is not inferred from one or more antecedent lines, but from a subderivation as a whole. The annotation is the range of lines occupied by the subderivation and the abbreviation CI. Note that the antecedent subderivation can consist of a single line serving both as the assumed φ and the derived ψ as in the following derivation of

$\therefore$ P → P

1.	P	Assumption
2.	P → P	

Unlike previously introduced inference rules, Conditional Introduction cannot be justified by a truth table. Rather it is justified by the Deduction Principle introduced at Properties of Sentential Connectives[1].

14.2 Negations

14.2.1 Example derivation

To illustrate Negation Introduction, we will provide a derivation for the argument

$\neg(\mathrm{P} \rightarrow \mathrm{Q}) \rightarrow \mathrm{R},\ \mathrm{P} \land \neg\mathrm{Q} \quad \therefore \quad \mathrm{Q} \lor \mathrm{R}$

1 Chapter 8.4.1 on page 87

1.	$\neg(P \to Q) \to R$	Premise
2.	$P \wedge \neg Q$	Premise

3. $P \to Q$
4. P
5. Q
6. $\neg Q$

Lines 3 through 6 constitute a subderivation. It starts by assuming the desired formula's opposite and ends by assuming a contradiction (a formula and its negation). As before, there are two vertical fences between the line numbers and the formulae to set it off from the rest of the derivation and to indicate its subordinate status. And the horizontal fence under line 3 again separates the assumption from the rest of the subderivation. Line 7, which follows from the entire subderivation, is the application of Negation Introduction.

At line 9, note that the annotation '5 DI' would be incorrect. Although inferring $Q \vee R$ from Q is valid by DI, line 5 is no longer active when we get to line 9. Thus we are not allowed to derive anything from line 5 at that point.

14.2.2 The Negation Introduction rule

Negation Introduction (NI)

$$\frac{\textit{Assume } \varphi\textit{, derive } \psi \textit{ and } \neg\psi}{\neg\varphi}$$

The consequent line is inferred from the whole subderivation. The annotation is the range of lines occupied by the subderivation and the abbreviation is NI. Negation Introduction sometimes goes by the Latin name *Reductio ad Absurdum* or sometimes by *Proof by Contradiction*.

Like Conditional Introduction, Negation Introduction cannot be justified by a truth table. Rather it is justified by the Reductio Principle introduced at Properties of Sentential Connectives[2].

14.2.3 Another example derivation

To illustrate Negation Elimination, we will provide a derivation for the argument

$\neg(\neg P \vee \neg Q) \wedge R \quad \therefore \quad P$

2 Chapter 8.4.1 on page 87

1.	$\neg(\neg P \vee \neg Q) \wedge R$	Premise
2.	$\neg Q$	
3.	$\neg P \vee \neg Q$	
4.	$\neg(\neg P \vee \neg Q)$	

Lines 2 through 4 constitute a subderivation. As in the previous example, it starts by assuming the desired formula's opposite and ends by assuming a contradiction (a formula and its negation). Line 5, which follows from the entire subderivation, is the application of Negation Elimination.

14.2.4 The Negation Elimination rule

Negation Elimination (NE)

$$\frac{\textit{Assume } \neg\varphi, \textit{ derive } \psi \textit{ and } \neg\psi}{\varphi}$$

The consequent line is inferred from the whole subderivation. The annotation is the range of lines occupied by the subderivation and the abbreviation is NE. Like Negation Introduction, Negation Elimination sometimes goes by the Latin name *Reductio ad Absurdum* or sometimes by *Proof by Contradiction.*

Like Negation Introduction, Negation Elimination is justified by the Reductio Principle introduced at Properties of Sentential Connectives[3]. This rule's place in the Introduction/Elimination naming convention is somewhat more awkward than for the other rules. Unlike the other elimination rules, the negation that gets eliminated by this rule does not occur in an already derived line. Rather the eliminated negation occurs in the assumption of the subderivation.

14.3 Terminology

The inference rules introduced in this module, Conditional Introduction and Negation Introduction, are discharge rules. For lack of a better term, we can call the inference rules introduced in Inference Rules[4] 'standard rules'. A *standard rule* is an inference rule whose antecedent is a set of lines. A *discharge rule* is an inference rule whose antecedent is a subderivation.

The *depth* of a line in a derivation is the number of fences standing between the line number and the formula. All lines of a derivation have a depth of at least one. Each temporary assumption increases the depth by one. Each discharge rule decreases the depth by one.

An *active line* is a line that is available for use as an antecedent line for a standard inference rule. In particular, it is a line whose depth is less than or equal to the depth of the current line. An inactive line is a line that is not active.

A discharge rule is said to *discharge* an assumption. It makes all lines in its antecedent subderivation inactive.

3 Chapter 8.4.1 on page 87
4 Chapter 11.3 on page 109

15 Constructing a Complex Derivation

15.1 An example derivation

Subderivations can be nested. For an example, we provide a derivation for the argument

$$\mathrm{P} \wedge \mathrm{R} \rightarrow \mathrm{T},\ \ \mathrm{S} \wedge \neg\mathrm{T},\ \ \mathrm{S} \rightarrow \neg\mathrm{Q} \quad \therefore \quad \mathrm{P} \vee \mathrm{Q} \rightarrow \neg\mathrm{R}$$

We begin with the premises and then assume the antecedent of the conclusion.

Note. Each time we begin a new subderivation and enter a temporary assumption, there is a specific formula we are hoping to derive when it comes time to end the derivation and discharge the assumption. To make things easier to follow, we will add this formula to the annotation of the assumption. That formula will not officially be part of the annotation and does not affect the correctness of the derivation. Instead, it will serve as an informal reminder to ourselves noting where we are going.

1.	$P \wedge R \to T$	Premise
2.	$S \wedge \neg T$	Premise
3.	$S \to \neg Q$	Premise

4. $P \vee Q$

This starts a subderivation to derive the argument's conclusion. Now we will try a Disjunction Elimination (DE) to derive its consequent:

$$\neg R$$

This will require the showing two conditionals we need for the antecedent lines of a DE, namely:

$$P \rightarrow \neg R$$

and

$$Q \rightarrow \neg R$$

We begin with the first of these conditionals.

5. P Assumption [P → ¬R]

6.

This subderivation is easily finished.

7.	$P \wedge R$	5, 6 KI
8.	T	1, 7 CE
9.	$\neg T$	2 KE

Now we are ready to discharge the two assumptions at Lines 5 and 6.

10. $\neg R$ 6–9 NI

11. $P \rightarrow \neg$

Now it's time for the second conditional needed for our DE planned back at Line 4. We begin.

12. Q Assumption $[Q \rightarrow \neg R]$

13.

14.

15.

Note that we have a contradiction between Lines 12 and 15. But line 12 is in the wrong place. We need it in the same subderivation as Line 15. A silly trick at Lines 16 and 17 below will accomplish that. Then the assumptions at Lines 12 and 13 can be discharged.

16.	$Q \wedge Q$	12, 12 KI	
17.	Q	16 KE	
			18.

Finally, with Lines 4, 11, and 19, we can perform the DE we've been wanting since Line 4.

20. ¬R 4, 11, 19 DE

Now to finish the derivation by discharging the assumption at Line 4.

21. $P \vee Q \rightarrow \neg R$ 4–20 CI

15.2 The complete derivation

Here is the completed derivation.

1.	$P \wedge R \rightarrow T$	Premise
2.	$S \wedge \neg T$	Premise
3.	$S \rightarrow \neg Q$	Premise

4. $P \vee Q$

16 Theorems

A *theorem* is a formula for which a zero-premise derivation has been provided. We will keep a numbered list of proved theorems. In the derivations that follow, we will continue our informal convention of adding a formula to the annotations of assumptions, in particular the formula we hope to derive by means of the newly started subderivation.

16.1 An example

You may remember from Constructing a Complex Derivation[1] that we had to employ a silly trick to copy a formula into the proper subderivation (Lines 16 and 17). We can prove a theorem that will help us avoid such obnoxiousness.

T1. $\mathrm{P} \rightarrow \mathrm{P}$

1 Chapter 14.3 on page 137

1. P Assumption $[P \to P]$
2. $\vdash P \to P$

Derivations can be abbreviated by allowing a line to be entered whose formula is a substitution instance of a previously proved theorem. The annotation will be 'Tn ***where*n *is the number of the theorem. Although we won't require it officially, we will also show the substitution, if any, in the annotation (see Line 3 in the derivation below). The proof of the next theorem will use*T1.**

T2. $Q \rightarrow (P \rightarrow Q)$

1. Q Assumption $[Q \to (P \to Q)]$
2. P
3. $Q \to Q$
4. Q

16.2 Justification: Converting to unabbreviated derivation

We need to justify using theorems in derivations in this way. To do that, we show how to produce a correct, unabbreviated derivation of **T2**, one without citing the theorem we used in its abbreviated proof.

Observe that when we entered Line 3 into our derivation of **T2**, we substituted Q for P in **T1**. Suppose you were to apply the same substitution on each line of our proof for **T1**. You would then end up with the following equally correct derivation.

1.	Q	Assumption	$[Q \rightarrow Q]$
2.	$Q \rightarrow Q$		

Suppose now you were to replace Line 3 of our proof for **T2** with this derivation. You would need to adjust the line numbers so that you would have only one line per line number. You would also need to adjust the annotations so the line numbers they would continue to refer correctly. But, with these adjustments, you would end up with the following correct unabbreviated derivation of **T2**.

1. Q Assumption $[Q \to (P \to Q)]$
2. P

Thus we see that entering a previously proved theorem into a derivation is simply an abbreviation for including that theorem's proof into a derivation. The instructions above for unabbreviating a derivation could be made more general and more rigorous, but we will leave them in this informal state. Having instructions for generating a correct unabbreviated derivation justifies entering previously proved theorems into derivations.

16.3 Additional theorems

Additional theorems will be introduced over the next two modules.

17 Derived Inference Rules

This page introduces the notion of a **derived inference rule** and provides a few such rules.

17.1 Deriving inference rules

17.1.1 The basics

Now we can carry the abbreviation a step further. A *derived inference rule* is an inference rule not given to us as part of the derivation system but which constitutes an abbreviation using a previously proved theorem. In particular, suppose we have proved a particular theorem. In this theorem, uniformly replace each sentence letter with a distinct Greek letter. Suppose the result has the following form. [Comment: This and what follows seems to me to be potentially confusing to students. The intention stated in earlier sections to avoid metatheory makes for a problem here as one really should know about the Deduction Theorem for this to make more sense.]

$$\varphi_1 \land \varphi_2 \land \ \ldots\ \varphi_n \rightarrow \psi$$

We may then introduce a derived inference rule having the form

$$\varphi_1$$

$$\varphi_2$$

$$\ldots$$

$$\underline{\varphi_n}$$

$$\psi$$

An application of the derived rule can be eliminated by replacing it with (i) the previously proved theorem, (ii) enough applications of Conjunction Introduction (KI) to build up the theorem's antecedent, and (iii) an application of Conditional Elimination (CE) to obtain the theorem's consequent. The previously proved theorem can then be eliminated as described above. That would leave you with an unabbreviated derivation.

Please remember, removing the abbreviations from a derivation is not desirable. It will make the derivation more complicated and harder to read. Rather, the fact that a derivation *could* be unabbreviated if so desired (even though we don't really desire it) is what justifies the abbreviation, what permits us to employ the abbreviation in the first place.

17.1.2 Repetition

Our first derived inference rule will be based on **T1**, which is

$$\mathrm{P} \rightarrow \mathrm{P}$$

Replace the sentence letters with Greek letters, and we get:

$$\varphi \rightarrow \varphi$$

We now generate the derived inference rule:

Repetition (R)

$$\underline{\varphi}$$

$$\varphi$$

Now we can show how this rule could have simplified our proof of **T2**.

1. Q Assumption $[Q \to (P \to Q)]$
2. P
3. Q

While this is only one line shorter than our original proof of **T2**, it is less obnoxious. We can use an inference rule instead of a silly trick. As a result, the derivation is easier to read and understand (not to mention easier to produce).

17.2 Double negation rules

The next two theorems—and the derived rules based on them—exploit the equivalence between a doubly negated formula and the unnegated formula.

17.2.1 Double Negation Introduction

T3. $\mathrm{P} \rightarrow \neg\neg \mathrm{P}$

1.	P	Assumption	$[P \rightarrow \neg\neg P]$
2.	$\neg P$		
3.	P		

T3 justifies the following rule.

Double Negation Introduction (DNI)

$$\frac{\varphi}{\neg\neg\varphi}$$

17.2.2 Double Negation Elimination

T4. $\neg\neg \mathrm{P} \to \mathrm{P}$

1. $\neg\neg$P Assumption [$\neg\neg$P $\rightarrow$ P]
2. $\neg$P
3. $\neg\neg$P

T4 justifies the following rule.

Double Negation Elimination (DNE)

$$\frac{\neg\neg\varphi}{\varphi}$$

17.3 Additional derived rules

17.3.1 Contradiction

$$\textbf{T5}. \quad \mathrm{P} \wedge \neg \mathrm{P} \rightarrow \mathrm{Q}$$

1. $P \wedge \neg P$ Assumption $[P \wedge \neg P \rightarrow Q]$

2. $\neg Q$

3. P

4. $\neg P$

Our next rule is based on **T5**.

Contradiction (Contradiction)

φ

$\underline{\neg\varphi}$

ψ

This rule is occasionally useful when you have derived a contradiction but the discharge rule you want is not NI or NE. This then avoids a completely trivial subderivation. The rule of Contradiction will be used in the proof of the next theorem.

17.3.2 Conditional Addition

T6. $\neg \mathrm{P} \rightarrow (\mathrm{P} \rightarrow \mathrm{Q})$

1. $\neg P$ Assumption $[\neg P \to (P \to Q)]$

2. P

3. Q

On the basis of **T2** and **T6**, we introduce the following derived rule.

Conditional Addition, Form I (CAdd)

$$\frac{\psi}{(\varphi \rightarrow \psi)}$$

Conditional Addition, Form II (CAdd)

$$\frac{\neg\psi}{(\psi \rightarrow \varphi)}$$

The name 'Conditional Addition' is not in common use. It is based on the traditional name for Disjunction Introduction, namely 'Addition'. This rule does not provide a general means of introducing a conditional. This is because the antecedent line you would need is not always derivable. However, when the antecedent line just happens to be easily available, then applying this rule is simpler than producing the subderivation needed for a Conditional Introduction.

17.3.3 Modus Tollens

$$\textbf{T7}. \quad (\mathrm{P} \rightarrow \mathrm{Q}) \land \neg \mathrm{Q} \rightarrow \neg \mathrm{P}$$

1.	$(P \to Q) \land \neg Q$	Assumption	$[(P \to Q) \land \neg Q \to \neg P]$

2\. P
3\. $P \to Q$
4\. Q
5\. $\neg Q$

Now we use **T7** to justify the following rule.

Modus Tollens (MT)

$$(\varphi \to \psi)$$

$$\underline{\neg\psi \quad\quad}$$

$$\neg\varphi$$

Modus Tollens is also sometimes known as 'Denying the Consequent'. Note that the following is **not** an instance of Modus Tollens, at least as defined above.

$$\neg P \to \neg Q$$

$$\underline{Q \quad\quad\quad}$$

$$P$$

The premise lines of Modus Tollens are a conditional and the negation of its consequent. The premise lines of this inference are a conditional and the **opposite** of its consequent, but not the **negation** of its consequent. The desired inference here needs to be derived as below.

1.	$\neg P \rightarrow \neg Q$	Premise
2.	Q	Premise
3.	$\neg\neg Q$	2 DNI
4.	$\neg\neg P$	1, 3 CE
5.	P	4 DNE

Of course, it is possible to prove as a theorem:

$$(\neg P \rightarrow \neg Q) \wedge Q \rightarrow P\ .$$

Then you can add a new inference rule—or, more likeley, a new form of Modus Tollens—on the basis of this theorem. However, we won't do that here.

17.4 Additional theorems

The derived rules given so far are quite useful for eliminating frequently used bits of obnoxiousness in our derivations. They will help to make your derivations easier to generate and also more readable. However, because they are indeed derived rules, they are not strictly required but rather are theoretically dispensable.

A number of other theorems and derived rules could usefully be added. We list here some useful theorems but leave their proofs and the definition of their associated derived inference rules to the reader. If you construct many derivations, you may want to maintain your own personal list that you find useful.

17.4.1 Theorems with biconditionals

T8. $(P \leftrightarrow Q) \wedge \neg P \rightarrow \neg Q$

T9. $(P \leftrightarrow Q) \wedge \neg Q \rightarrow \neg P$

T10. $P \wedge Q \rightarrow (P \leftrightarrow Q)$

T11. $\neg P \wedge \neg Q \rightarrow (P \leftrightarrow Q)$

17.4.2 Theorems with negations

T12. $\neg(P \rightarrow Q) \rightarrow P \wedge \neg Q$

T13. $\neg(P \leftrightarrow Q) \rightarrow (P \leftrightarrow \neg Q)$

T14. $\neg(P \vee Q) \rightarrow \neg P \wedge \neg Q$

T15. $\neg(P \wedge Q) \rightarrow \neg P \vee \neg Q$

18 Disjunctions in Derivations

Disjunctions in derivations are, as the current inference rules stand, difficult to deal with. Using an already derived disjunction by applying Disjunction Elimination (DE) is not too bad, but there is an easier to use alternative. Deriving a disjunction in the first place is more difficult. Our Disjunction Introduction (DI) rule turns out to be a rather anemic tool for this task. In this module, we introduce derived rules which provide alternative methods for dealing with disjunctions in derivations.

18.1 Using already derived disjunctions

18.1.1 Modus Tollendo Ponens

We start with a new (to be) derived rule of inference. This will provide a useful alternative to Disjunction Elimination (DE).

Modus Tollendo Ponens, Form I (MTP)

$$\frac{\begin{array}{c}(\varphi \vee \psi)\\ \neg\varphi\end{array}}{\psi}$$

Modus Tollendo Ponens, Form II (MTP)

$$\frac{\begin{array}{c}(\varphi \vee \psi)\\ \neg\psi\end{array}}{\varphi}$$

Modus Tollendo Ponens is sometimes known as Disjunctive Syllogism and occasionally as the Rule of the Dog.

18.1.2 Supporting theorems

This new rule requires the following two supporting theorems.

T16. $(\mathrm{P} \vee \mathrm{Q}) \wedge \neg \mathrm{P} \rightarrow \mathrm{Q}$

7. $(P \vee Q) \wedge \neg P \to Q$

1.	$(P \vee Q) \wedge \neg P$	Assumption	$[(P \vee Q) \wedge \neg P \to Q]$
2.	$P \vee Q$	1 KE	
3.	$\neg P$	1 KE	
4.	$P \to Q$	3 CAdd	
5.	$Q \to Q$	T1 $[\mathbf{P}/\mathbf{Q}]$	
6.	Q	2, 4, 5 DE	

T17. $(P \lor Q) \land \neg Q \to P$

7. $(P \vee Q) \wedge \neg Q \to P$

1.	$(P \vee Q) \wedge \neg Q$	Assumption	$[(P \vee Q) \wedge \neg Q \to P]$
2.	$P \vee Q$	1 KE	
3.	$\neg Q$	1 KE	
4.	$Q \to P$	3 CAdd	
5.	$P \to P$	T1	
6.	P	2, 4, 5 DE	

18.1.3 Example derivation

For an example using MTP, we redo the example derivation from Constructing a Complex Derivation[1].

$$P \land R \rightarrow T, \quad S \land \neg T, \quad S \rightarrow \neg Q \quad \therefore \quad P \lor Q \rightarrow \neg R$$

1 Chapter 14.3 on page 137

1.	$P \wedge R \rightarrow T$	Premise
2.	$S \wedge \neg T$	Premise
3.	$S \rightarrow \neg Q$	Premise

4. $P \vee Q$

After Line 4, we did not bother with subderivations for deriving the antecedent lines needed for DE. Instead, we went straight to a subderivation for the conclusion's consequent. At line 8, we applied MTP.

18.2 Deriving disjunctions

18.2.1 Conditional Disjunction

The next derived rule significantly reduces the labor of deriving disjunctions, thus providing a useful alternative to Disjunction Introduction (DI).

Conditional Disjunction (CDJ)

$$\frac{(\neg\varphi \to \psi)}{(\varphi \vee \psi)}$$

18.2.2 Supporting theorem

$$\textbf{T18}. \quad (\neg \mathrm{P} \to \mathrm{Q}) \to \mathrm{P} \vee \mathrm{Q}$$

1. $\neg P \to Q$ Assumption $[(\neg P \to Q) \to P \vee Q]$

2. $\neg(P \vee Q)$

18.2.3 Example derivation

This derivation will make use of **T12** (introduced at Derived Inference Rules[2]) even though its proof was left to the reader as an exercise. The correctness the following derivation, particularly Line 2, assumes that you have indeed proved **T12**.

$\therefore \quad (P \rightarrow Q) \lor (Q \rightarrow R)$

2 Chapter 16.3 on page 167

1.	$\neg(P \to Q)$	Assumption	$[\neg(P \to Q) \to (Q \to R)]$
2.	$\neg(P \to Q) \to P \wedge \neg Q$	T12	
3.	$P \wedge \neg Q$	1, 2 CE	
4.	$\neg Q$	3 KE	
5.	$Q \to R$	4 CAdd	

7.	$\neg(P \to Q) \to (Q \to R)$
8.	$(P \to Q) \vee (Q \to R)$

Here we attempted to derive the desired conditional by first deriving the antecedent line needed for CDJ.

3

Predicate Logic

3 https://en.wikibooks.org/wiki/Category%3A

19 Goals

Sentential logic[1] treated whole sentences and truth functional relations among them. **Predicate logic** treats more fine-grained logical features. This page will informally describe of the logic features of English captured by predicate logic.

19.1 Predicate logic goals

19.1.1 Predicates and terms

We distinguish between *predicates* and *terms*. The simplest terms are simple names such as 'one', 'Socrates', or 'Pegasus'. Simple predicates attribute properties to the object named. In

Socrates is bald.

'Socrates' is a name and 'is bald' is the predicate. We can informally exhibit its logical structure with

Bald(socrates)

There are also more complex terms such as '3 minus 2', 'the father of Socrates', and '3 divided by 0'. The logical structure of these can be exhibited informally by

minus(3, 2)

father(socrates)

quotient(3, 0)

These more complex terms can be used to construct sentences such as

3 minus 2 is even.

The father of Socrates is bald.

3 divided by zero is odd.

Their logical structure can be informally exhibited by

Even(minus(3, 2))

Bald(father(Socrates))

Odd(quotient(3, 0))

1 Chapter 1.3.3 on page 9

There are also complex predicates, often called *relations*. Sentences using such predicates include

3 is greater than 2

Socrates is a child of the father of Socrates and the mother of Socrates.

Pegasus kicked Bucephalus.

Their logical structures can be informally exhibited as

Greater_than(3, 2)

Child(socrates, father(socrates), mother(socrates))

Kicked(pegasus, bucephalus)

19.1.2 General statements

Names and other terms refer to specific objects. We can also speak generally of all objects or of some (at least one) objects. Some examples are:

All numbers are prime.

Some numbers are prime.

Some numbers are not prime.

No numbers are prime.

The logical structures of the first three can be informally exhibited as

All x (if Number(x), then Prime(x)).

Some x (Number(x) and Prime(x)).

Some x (Number(x) and not(Prime(x))).

The fourth can have its logical structure exhibited either as

All x (if Number(x), then not Prime(x).

or, equivalently, as

Not (some x (Number(x) and Prime(x))).

Note that we count

All unicorns have a horn

as trivially true because there are no unicorns. In addition, we take 'some' to mean *at least one* (not, as might be expected, *at least two*). Thus we take

Some prime numbers are even

to be true even though two is the only even prime.

19.1.3 More complexity

Variables such as 'x' and 'y' help us to keep the following straight.

- All x all y (if Person(x) and Person(y), then Loves(x, y))
- All x all y (if Person(y) and Person(x), then Loves(y, x))

  These are equivalent and both say that everybody loves everybody.

- Some x some y (Person(x) and Person(y) and Loves(x, y))
- Some x Some y (Person(y) and Person(x) and Loves(y, x))

  These are equivalent and both say that somebody loves someone.

- All x (if Person(x), then some y (Person(y) and Loves(x, y)))
- Some y (Person(y) and all x (if Person(x), then Loves(x, y)))

  The first says that everybody loves somebody (or other—they do not necessarily all love the same person). The second says that somebody is loved by everybody. Thus the second, but not the first, requires a universally loved person.

- All x (if Person(x), then some y (Person(y) and Loves(y, x)))
- Some y (Person(y) and all x (if Person(x), then Loves(y, x)))

  The first says that that everybody is loved by somebody (or other—not necessarily all loved by the same person). The second says that the somebody loves everybody. Thus the second, but not the first, requires a universal lover.

19.1.4 Domains

By convention, we can temporarily limit the range of the objects being considered. For example, a policy of speaking only about people might be conventionally adopted. Such a convention would allow us to hope optimistically that

All x some y Loves(y, x)

without hoping that cups and saucers are thereby loved. The truth or falsity of a sentence can be evaluated within the context of such a convention or policy. In a different context, a different convention might prove more convenient. For example, the policy of only speaking about people would prevent us from saying

Loves(alexander_the_great, beaucephalus)

However, widening the domain to include both horses and people does allow us to say this.

We call the range of objects under discussion the *domain*, or sometimes the *domain of discourse*. A sentence that is true in the context of one domain may be false in the context of another.

19.2 Limits

There are two limits on predicate logic as conventionally developed and indeed as will be developed here.

- First, predicate logic assumes that at least one thing exists. We cannot, for example, limit the domain of discourse to unicorns.
- Second names and complex terms must refer to objects in the domain. Thus we cannot use such terms as

Pegasus

3 divided by 0

In addition, if we limit the domain to natural numbers, we cannot use a term such as

2 minus 3

The loss of '3 divided by 0' in turn requires predicate logic to avoid forming any term with 'divided by'. The loss of '2 minus 3' requires predicate logic to avoid forming any term with 'minus' except in the context of domains including negative numbers.

The significance of these limits is controversial. Free Logic[2] attempts to avoid such limits. It appears that there is a Stanford Encyclopedia of Philosophy[3] entry on free logic in preparation.

2 https://en.wikipedia.org/wiki/Free%20logic
3 http://plato.stanford.edu/

20 The Predicate Language

This page informally describes our **predicate language** which we name $\mathcal{L}_{\mathcal{P}}$. A more formal description will be given in subsequent pages.

20.1 Language components

Use of $\mathcal{L}_{\mathcal{P}}$ occurs in the context of a *domain* of objects. Ascribing a property to everything is interpreted as ascribing it just to everything in the domain.

20.1.1 Terms

Variables will be lower case letters $\boldsymbol{n}$ through $\boldsymbol{z}$. Variables serve roughly as placeholders or perhaps pronouns, particularly in general statements about all or some objects. A variable could serve as the 'it' in 'For any number, if it is even then it is not odd'.

Operation letters of zero or more places will consist of lower case letters $\boldsymbol{a}$ through $\boldsymbol{m}$. Since the same letters are used for operation letters of any number of places, some disambiguation will be necessary. For now, just let the context decide the number of places. Note, though, that variables always have zero places.

Terms can be one of the following.

- A variable.
- A zero-place operation letter.
- An n-place operation letter (n being 1 or greater) followed by a parenthesised list of n terms. Examples include

$$f(c)$$

$$g(x, y)$$

Names are terms in which no variables occur. Names name. In particular, they name objects in the domain. Variables, and so terms containing variables (i.e., terms that are not names), do not name.

For the remainder of this page, assume the following translations.

$$c : \text{ Cain}$$

$$f(x) \ : \ \text{the father of } x$$

$$g(x,y) \ : \ \text{the greater of } x \text{ and } y$$

With the right set of characters in the domain, biblical tradition has

$$c$$

$$f(c)$$

naming Cain and Adam respectively. The term

$$g(x,y)$$

doesn't name anything. However, if we abuse $\mathcal{L}_\mathcal{P}$ a bit by permitting numerals to serve as zero place operation letters, then the terms

$$g(7,3)$$

$$g(3,3)$$

name 7 and 3 respectively (assuming that 7 and 3 are in the domain).

20.1.2 Primitive formulae

Predicate letters of zero or more places will consist of capital letters **A** through **Z**. The same letters will be used for predicate letters of any number of places, so, as with operation letters, we will need to disambiguate. Again as with operation letters, we will let context decide the number of places for now. Notice that zero-place predicate letters are sentence letters we are familiar with from sentential logic[1].

Primitive formulae can be one of the following.

- A zero-place predicate letter (that is, a sentence letter).
- An n-place predicate letter (n being 1 or greater) followed by a parenthesised list of n terms. Examples include

$$\mathrm{P}$$

1 Chapter 1.3.3 on page 9

$$\mathrm{F}(f(c))$$

$$\mathrm{G}(g(x,y),u,v)$$

If P translates 'Snow is white', then it is true. However, it is false if it translates 'Snow is blue'.

Suppose we add

$$\mathrm{F}(x) \ : \ \ x \text{ is bald}$$

$$b \ : \ \ \text{Yul Brenner}$$

$$k \ : \ \ \text{Don King}$$

to the translations above. Then

$$\mathrm{F}(f(c))$$

is true or false according with whether Adam was bald. We say that F is true of all bald things and false of all non-bald things. Thus the first of

$$\mathrm{F}(b)$$

$$\mathrm{F}(k)$$

is true while the second is false. For confirmation of these truth evaluations, see pictures accompanying the Wikipedia[2] articles on Yul Brenner[3] and Don King[4]. (Note. At one point, the picture of Don King had been removed from the Wikipedia article. However, he rather famously is not bald.)

Now add

$$\mathrm{G}(x,y,z) \ : \ \ x \text{ is between } y \text{ and } z$$

to the translations above. Then

$$\mathrm{G}(g(x,y),u,v)$$

2 `https://en.wikipedia.org/wiki/Main%20Page`
3 `https://en.wikipedia.org/wiki/Yul%20Brenner`
4 `https://en.wikipedia.org/wiki/Don%20King%20%28boxing%20promoter%29`

is neither true nor false because, as above,

$$g(x,y)$$

does not name anything. For that matter, neither do the variables u or v. But if we again abuse $\mathcal{L}_\mathcal{P}$ a bit by permitting numerals to serve as zero place operation letters, then the first of

$$\mathrm{G}(g(3,3),1,4)$$

$$\mathrm{G}(g(7,3),1,4)$$

is true while the second is false.

20.1.3 Sentential connectives

The predicate language $\mathcal{L}_\mathcal{P}$ will use sentential connectives just as they were used in the sentential language $\mathcal{L}_\mathcal{S}$. These were:

$$\neg,\ \wedge,\ \vee,\ \rightarrow,\ \text{and}\ \leftrightarrow$$

Using the translations already set above (together with letting numerals be zero-place operation letters),

$$\mathrm{F}(f(d)) \vee \mathrm{G}(g(7,3),1,4)$$

is true while

$$\mathrm{F}(f(d)) \wedge \mathrm{G}(g(7,3),1,4)$$

is false.

20.1.4 Quantifiers

Quantifiers are special symbols that allow us to construct general sentences which are about all thing or about some (at least one) things.

Universal quantifier: $\forall$

- $\forall x$ translates to English as 'for all x.
- $\forall x\ \mathrm{F}(x)$ is called a *universal generalization.*
- $\forall x\ \mathrm{F}(x)$ is true if $\mathrm{F}(x)$ is true of all objects in the domain. Roughly speaking, it is true if

$$F(a_1) \wedge F(a_2) \wedge ... \wedge F(a_n)$$

where each (a_i) names an object in the domain and all objects in the domain are named. This is only a rough characterization, however. First, we do not require that all objects in the domain have a name in the predicate languange. Second, we allow there to be infinitely many objects in the domain but do not allow infinitely long sentences.

- Some authors use (x) instead of $\forall x$. This notation is semi-obsolete and is becoming ever less frequent.

Existential quantifier: $\exists$

- $\exists x$ translates to English as 'there exists an x' or, perhaps a bit more clearly, 'there exists at least one x'.
- $\exists x\ F(x)$ is called an *existential generalization.*
- $\exists x\ F(x)$ is true if $F(x)$ is true of at least one object in the domain. Roughly speaking, it is true if

$$F(a_1) \vee F(a_2) \vee ... \vee F(a_n)$$

where each (a_i) names an object in the domain and all objects in the domain are named. This is only a rough characterization, however. First, we do not require that all objects in the domain have a name in the predicate languange. Second, we allow there to be infinitely many objects in the domain but do not allow infinitely long sentences.

20.2 Translation

Using the translation scheme

$$F(x) \ : \quad x \text{ is a number}$$

$$G(x) \ : \quad x \text{ is prime}$$

we translate as follows.

All numbers are prime.

$\forall x(Fx \rightarrow Gx)$

Some numbers are prime.

$\exists x(Fx \wedge Gx)$

No numbers are prime. *(two equivalent alternatives are given)*

$\forall x(Fx \rightarrow \neg Gx)$

$\neg\exists x(Fx \wedge Gx)$

Some numbers are not prime.

$\exists x(Fx \wedge \neg Gx)$

Now using the translation scheme

$$\mathrm{P}(x) \ : \ \ x \text{ is a person}$$

$$\mathrm{L}(x,y) \ : \ \ x \text{ loves } y$$

$$g \ : \ \ \text{George}$$

$$m \ : \ \ \text{Martha}$$

we can translate as follows.

George loves Martha.

$\mathrm{L}(g,m)$

Martha loves George.

$\mathrm{L}(m,g)$

George and Martha love each other.

$\mathrm{L}(g,m) \wedge \mathrm{L}(m,g)$

We can further translate as follows.

Everybody loves everybody. *(the second alternative assumes only persons in the domain)*

$\forall x(\mathrm{P}(x) \rightarrow \forall y(\mathrm{P}(y) \rightarrow \mathrm{L}(x,y)))$

$\forall x \forall y \mathrm{L}(x,y)$

Somebody loves somebody. *(the second alternative assumes only persons in the domain)*

$\exists x(\mathrm{P}(x) \wedge \exists y(\mathrm{P}(y) \wedge \mathrm{L}(x,y)))$

$\exists x \exists y \mathrm{L}(x,y)$

Everybody loves somebody (or other). *(the second alternative assumes only persons in the domain)*

$\forall x(\mathrm{P}(x) \rightarrow \exists y(\mathrm{P}(y) \wedge \mathrm{L}(x,y)))$

$\forall x \exists y \mathrm{L}(x,y)$

Somebody is loved by everybody. *(the second alternative assumes only persons in the domain)*

$\exists y \forall x(\mathrm{P}(y) \wedge (\mathrm{P}(x) \rightarrow \mathrm{L}(x,y)))$

$\exists y \forall x \mathrm{L}(x,y)$

Everybody is loved by somebody (or other). *(the second alternative assumes only persons in the domain)*

$\forall x \exists y (\mathrm{P}(x) \to \mathrm{P}(y) \land \mathrm{L}(y, x))$

$\forall x \exists y \mathrm{L}(y, x)$

Somebody loves everybody. *(the second alternative assumes only persons in the domain)*

$\exists y \forall x (\mathrm{P}(y) \land (\mathrm{P}(x) \to \mathrm{L}(y, x)))$

$\exists y \forall x \mathrm{L}(y, x)$

21 Formal Syntax

In The Predicate Language[1], we informally described our sentential language. Here we give its **formal syntax** or grammar. We will call our language $\mathcal{L_P}$. This is an expansion of the sentential language $\mathcal{L_S}$ and will include $\mathcal{L_S}$ as a subset.

21.1 Vocabulary

- *Variables:* Lower case letters 'n'−'z' with a natural number subscript. Thus the variables are:

$$n_0,\ n_1,\ ...,\ o_0,\ o_1,\ ...,\ ...,\ z_0,\ z_1,\ ...$$

- *Operation letters:* Lower case letters 'a'−'m' with (1) a natural number superscript and (2) a natural number subscript.

$$a_0^0,\ a_1^0,\ ...,\ b_0^0,\ b_1^0,\ ...,\ ...,\ m_0^0,\ m_1^0,\ ...$$

$$a_0^1,\ a_1^1,\ ...,\ b_0^1,\ b_1^1,\ ...,\ ...,\ m_0^1,\ m_1^1,\ ...$$

...

A *constant symbol* is a zero-place operation letter. This piece of terminology is not completely standard.

- *Predicate letters:* Upper case letters 'A'−'Z' with (1) a natural number superscript and (2) a natural number subscript.

$$\mathrm{A}_0^0,\ \mathrm{A}_1^0,\ ...,\ \mathrm{B}_0^0,\ \mathrm{B}_1^0,\ ...,\ ...,\ \mathrm{Z}_0^0,\ \mathrm{Z}_1^0,\ ...$$

$$\mathrm{A}_0^1,\ \mathrm{A}_1^1,\ ...,\ \mathrm{B}_0^1,\ \mathrm{B}_1^1,\ ...,\ ...,\ \mathrm{Z}_0^1,\ \mathrm{Z}_1^1,\ ...$$

...

A *sentence letter* is a zero-place predicate letter.

1 Chapter 19.2 on page 200

- *Sentential connectives*:

$$\wedge, \vee, \neg, \rightarrow, \leftrightarrow$$

- *Quantifiers*:

$$\forall, \text{ and } \exists,$$

- Grouping symbols:

$$(\text{ and })$$

The superscripts on operation letters and predicate letters indicate the number of places and are important for formation rules. The subscripts on variables, operation letters, and predicate letters are to ensure an infinite supply of symbols in these classes. On a subsequent page we will abbreviate away most superscript use by letting the context make the number of places clear. We will also abbreviate away most subscript use by letting a symbol without a subscript abbreviate one with the subscript '0'. For now, though, we stick with the unabbreviated form.

The sentence letters of sentential logic are zero-place predicate letters, namely predicate letters with the superscript '0'. The vocabulary of $\mathcal{L}_\mathcal{S}$, the sentential logic formal language, includes zero-place predicate letters, sentential connectives, and grouping symbols.

21.2 Expressions

Any string of symbols from $\mathcal{L}_\mathcal{P}$ is an *expression*. Not all expressions are grammatically well formed. The primary well-formed expression is a formula. However, there are also well formed entities that are smaller than formulae, namely quantifier phrases and terms.

21.3 Formation rules

21.3.1 Quantifier phrases

A *quantifier phrase* is a quantifier followed by a variable. The following are examples:

$$\forall x_0$$

$$\exists y_{37}$$

21.3.2 Terms

An expression of $\mathcal{L}_{\mathcal{P}}$ is a *term* of $\mathcal{L}_{\mathcal{P}}$ if and only if it is constructed according to the following rules.

i. A variable is a term.

ii. A constant symbol (zero-place operation letter, i.e., an operation letter with the superscript '0') is a term.

iii. If ζ is an n-place operation letter (n greater than 0) and α_1, α_2, ..., α_n are terms, then

$$\zeta(\alpha_1, \alpha_2, ..., \alpha_n)$$

is a term.

A *name* is a term with no variables.

21.3.3 Formulae

An expression of $\mathcal{L}_{\mathcal{P}}$ is a *well-formed formula* of $\mathcal{L}_{\mathcal{P}}$ if and only if it is constructed according to the following rules.

i. A sentence letter (a zero-place predicate letter) is a well-formed formula.

ii. If π is an n-place predicate letter (n greater than 0) and α_1, α_2, ..., α_n are terms, then

$$\pi(\alpha_1, \alpha_2, ..., \alpha_n)$$

is a well-formed formula.

iii. If φ and ψ are well-formed formulae, then so are each of:

iii-a. $\neg\varphi$

iii-b. $(\varphi \wedge \psi)$

iii-c. $(\varphi \vee \psi)$

iii-d. $(\varphi \rightarrow \psi)$

iii-e. $(\varphi \leftrightarrow \psi)$

iv. If φ is a well-formed formula and α is a variable, then each of the following is a well-formed formula:

iv-a. $\forall\alpha\varphi$

iv-b. $\exists\alpha\varphi$

In general, we will use 'formula' as shorthand for 'well-formed formula'. We will see in the section Free and Bound Variables[2] that only some formulae are sentences.

21.4 Additional terminology

A few of these terms are repeated from above. All definitions from the sentential logic additional terminology[3] section apply here except the definitions of 'atomic formula' and 'molecular formula'. These latter two terms are redefined below.

A *constant symbol* is a zero-place operation letter. (Note that different authors will vary on this.)

A *name* is a term in which no variables occur. (Note that different authors will vary on this. Some use 'name' only for zero-place operation letters, and some prefer to avoid the word altogether.)

A *sentence letter* is a zero-place predicate letter.

The *universal quantifier* is the symbol $\forall$. The *existential quantifier* is the symbol $\exists$.

A *quantified formula* is a formula that begins with a left parenthesis followed by a quantifier. A *universal generalization* is a formula that begins with a left parenthesis followed by a universal quantifier. An *existential generalization* is a formula that begins with a left parenthesis followed by a existential quantifier.

An *atomic formula* is one formed solely by formula formation clause {i} or {ii}. Put another way, an atomic formula is one in which no sentential connectives or quantifiers occur. A *molecular formula* is one that is not atomic. Thus a molecular formula has at least one occurrence of either a sentential connective or a quantifier. *(Revised from sentential logic.)*

A *prime formula* is a formula that is either an atomic formula or a quantified formula. A *non-prime formula* is one that is not prime. (Note that this is not entirely standard terminology. It has been used this way by some authors, but not often.)

The *main operator* of a molecular formula is the last occurrence of a sentential connective or quantifier added when the formula was constructed according to the rules above. If the main operator is a sentential connective, then it is also called the 'main connective' (as was done in the sentential language $\mathcal{L_S}$). However, there is a change as we move to $\mathcal{L_P}$. In predicate logic it is no longer true that all molecular formulae have a main connective. Some main operators are now quantifiers rather than sentential connectives.

2 Chapter 21.5 on page 214
3 Chapter 3.5 on page 19

21.5 Examples

$$(1) \quad f_0^2(g_2^2(x_0, f_0^2(a_0^0, c_2^0)), f_0^2(b_0^0, y_3))$$

By clause (i) in the definition of 'term', x_0 and y_3 are terms. Similarly, a_0^0, b_0^0, and c_2^0 are terms by clause (ii) of the definition of 'term'.

Next, by clause (iii) of the definition of 'term', the following two expressions are terms.

$$f_0^2(a_0^0, c_2^0)$$

$$f_0^2(b_0^0, y_3)$$

Then, by clause (iii) of the definition of 'term', the following is a term.

$$g_2^2(x_0, f_0^2(a_0^0, c_2^0))$$

Finally, (1) is a term by clause (iii) of the definition of 'term'. However, because it contains variables, it is not a name.

$$(2) \quad \mathrm{F}_0^1(f_0^2(g_2^2(x_0, f_0^2(a_0^0, c_2^0)), f_0^2(b_0^0, y_3)))$$

We already saw that (1) is a term. Thus, by clase (ii) of the definition of formula, (2) is a formula.

$$(3) \quad \exists y_0\,(\forall x_0\,(\mathrm{F}_0^2(x_0, y_0) \rightarrow \mathrm{G}_0^2(x_0, z_0)) \land \mathrm{H}_0^1(y_0))$$

By clause (i) in the definition of 'term', x_0, y_0, and z_0 are terms.

By clause (ii) of the definition for 'formula', the following are formulae.

$$\mathrm{F}_0^2(x_0, y_0)$$

$$\mathrm{G}_0^2(x_0, z_0)$$

$$\mathrm{H}_0^1(y_0)$$

By clause (iii-d) of the definition for 'formula', the following is a formula.

$$(\mathrm{F}_0^2(x_0, y_0) \rightarrow \mathrm{G}_0^2(x_0, z_0))$$

By clause (iv-a) of the definition for 'formula', the following is a formula.

$$\forall x_0\,(\mathrm{F}_0^2(x_0, y_0) \rightarrow \mathrm{G}_0^2(x_0, z_0))$$

By clause (iii-b) of the definition for 'formula', the following is a formula.

$$(\forall x_0\,(\mathrm{F}_0^2(x_0, y_0) \rightarrow \mathrm{G}_0^2(x_0, z_0)) \wedge \mathrm{H}_0^1(y_0))$$

Finally, by clause (iv-b) of the definition for 'formula', (3) is a formula.

22 Free and Bound Variables

22.1 Informal notions

The two English sentences,

If Socrates is a person, then Socrates is mortal,

if Aristotle is a person, then Aristotle is mortal,

are both true. However, outside any context supplying a reference for 'it',

(1) If it is a person, then it is mortal,

is neither true nor false. 'It' is not a name, but rather an empty placeholder. 'It' can refer to an object by picking up its reference from the surrounding context. But without such context, there is no reference and no truth or falsity. The same applies to the variable 'x' in

(2) If x is a person, then x is mortal.

This situation changes with the two sentences:

(3) For any object, if it is a person, then it is mortal,

(4) For any object x, if x is a person, then x is mortal.

Neither the occurrences of 'it' nor the occurrences of 'x' in these sentences refer to specific objects as with 'Socrates' or 'Aristotle'. But (3) and (4) are nonetheless true. (3) is true if and only if:

(5) Replacing both occurrences of 'it' in (3) with a reference to any object whatsoever (the same object both times) yields a true result.

But (5) is true and so is (3). Similarly, (4) is true if and only if:

(6) Replacing both occurrences of 'x' in (4) with a reference to any object whatsoever (the same object both times) yields a true result.

But (3) is true and so is (4). We can call the occurrences of 'it' in (1) free and the occurrences of 'it' in (3) bound. Indeed, the occurrences of 'it' in (3) are bound by the phrase 'for any'. Similarly, the occurrences 'x' in (2) are free while those in (4) are bound. Indeed, the occurrences of 'x' in (4) are bound by the phrase 'for any'.

22.2 Formal definitions

An occurrence of a variable α is bound in φ if that occurrence of α stands within a subformula of φ having one of the two forms:

$$\forall\alpha\,\psi\ ,$$

$$\exists\alpha\,\psi\ .$$

Consider the formula

$$(7)\quad (\exists x_0\,\mathrm{F}^1_0(x_0) \rightarrow \forall y_0\,\mathrm{F}^1_0(y_0))\ .$$

Both instances of x_0 are bound in (7) because they stand within the subformula

$$\exists x_0\,\mathrm{F}^1_0(x_0)\ .$$

Similarly, both instances of y_0 are bound in (7) because they stand within the subformula

$$\forall y_0\,\mathrm{F}^1_0(y_0)\ .$$

An occurrence of a variable α *is free* in φ if and only if α is not bound in φ. The occurrences of both x_0 and y_0 in

$$(8)\quad (\mathrm{F}^1_0(x_0) \rightarrow \mathrm{G}^1_0(y_0))$$

are free in (8) because neither is bound in (8).

We say that *an occurrence of a variable* α *is bound in by a particular occurrence* of $\forall$ if that occurrence is also the first (and perhaps only) symbol in the shortest subformula of φ having the form

$$\forall\alpha\,\psi\ .$$

Consider the formula

$$(9)\quad \forall x_0\,(\mathrm{F}^1_0(x_0) \rightarrow \forall x_0\,\mathrm{G}^1_0(x_0))\ .$$

The third and fourth occurrences of x_0 in (9) are bound by the second occurrence of $\forall$ in (9). However, they are not bound by the first occurrence of $\forall$ in (9). The occurrence of

$$(10)\quad \forall x_0\,\mathrm{G}^1_0(x_0)$$

in (9)—as well as the occurrence of (9) itself in (9)—are subformulae of (9) beginning with a quantifier. That is, both are subformula of (9) having the form

$$\forall\alpha\,\psi\ .$$

Both contain the second third and fourth occurrences of x_0 in (9). However, the occurrence of (10) in (9) is the *shortest* subformula of (9) that meets these conditions. That is, the occurrence of (10) in (9) is the shortest subformula of (9) that both (i) has this form and (ii) contains the third and fourth occurrences of x_0 in (9). Thus it is the second, not the first, occurrence of $\forall$ in (9) that binds the third and forth occurrences of x_0 in (9). The first occurrence of $\forall$ in (9) *does*, however, bind the first two occurrences of x_0 in (9).

We also say that *an occurrence of a variable* α *is bound in by a particular occurrence* of $\exists$ if that occurrence is also the first (and perhaps only) symbol in the shortest subformula of φ having the form

$$\exists \alpha \psi \ .$$

Finally, we say that *a variable* α (not a particular occurrence of it) *is bound (or free)* in a formula if the formula contains a bound (or free) occurrence of α. Thus x_0 is both bound and free in

$$(\forall x_0 \mathrm{F}^1_0(x_0) \rightarrow \mathrm{F}^1_0(x_0))$$

since this formula contains both bound and free occurrences of x_0. In particular, the first two occurrences of x_0 are bound and the last is free.

22.3 Sentences and formulae

A *sentence* is a formula with no free variables. Sentential logic had no variables at all, so all formulae of $\mathcal{L}_{\mathcal{S}}$ are also sentences of $\mathcal{L}_{\mathcal{S}}$. In predicate logic and its language $\mathcal{L}_{\mathcal{P}}$, however, we have formulae that are not sentences. All of (7), (8), (9), and (10) above are formulae. Of these, only (7), (9), and (10) are sentences. (8) is not a sentence because it contains free variables.

22.4 Examples

All occurrences of x_0 in

$$(11) \quad \forall x_0(\mathrm{F}^1_0(x_0) \rightarrow \mathrm{G}^2_0(x_0, y_0))$$

are bound in the formula. The lone occurrence of y_0 is free in the formula. (11) is a formula but not a sentence.

Only the first two occurrences of x_0 in

$$(12) \quad (\forall x_0 \mathrm{F}^1_0(x_0) \rightarrow \mathrm{G}^2_0(x_0, y_0))$$

are bound in the formula. The last occurrence of x_0 and the lone occurrence of y_0 in the formula are free in the formula. (12) is a formula but not a sentence.

All four occurrences of x_0 in

$$(13) \quad (\forall x_0 \mathrm{F}^1_0(x_0) \rightarrow \exists x_0 \mathrm{G}^2_0(x_0, y_0))$$

are bound. The first two are bound by the universal quantifier, the last two are bound by the existential quantifier. The lone occurrence of y_0 in the formula is free. (13) is a formula but not a sentence.

All three occurrences of x_0 in

$$(14) \quad \forall x_0(\mathrm{F}^1_0(x_0) \rightarrow \exists y_0 \mathrm{G}^2_0(x_0, y_0))$$

are bound by the universal quantifier. Both occurrences of y_0 in the formula are bound by the existential quantifier. (14) has no free variables and so is a sentence and as well as a formula.

23 Models

23.1 Interpretations

We said earlier[1] that the formal semantics for a formal language such as $\mathcal{L_S}$ (and now $\mathcal{L_P}$) goes in two parts.

- Rules for specifying an interpretation. An *interpretation* assigns semantic values to the non-logical symbols of a formal syntax. Just as a valuation was an interpretation for a sentential language, a *model* is an interpretation for a predicate language.
- Rules for assigning semantic values to larger expressions of the language. All formulae of the sentential language $\mathcal{L_S}$ are sentences. This enabled rules for assigning truth values directly to larger formulae. For the predicate language $\mathcal{L_P}$, the situation is more complex. Not all formulae of $\mathcal{L_P}$ are sentences. We will need to define the auxiliary notion *satisfaction* and use that when assigning truth values.

23.2 Models

A *model* is an interpretation for a predicate language. It consists of two parts: a domain and interpretation function. Along the way, we will progressively specify an example model $\mathfrak{M}$.

23.2.1 Domain

- A *domain* is a non-empty set.

Intuitively, the domain contains all the objects under current consideration. It contains all of the objects over which the quantifiers range. $\forall x$ is then interpreted as 'for any object x in the domain ...'; $\exists x$ is interpreted as 'there exists at least one object x in the domain such that ...'. Our predicate logic requires that the domain be non-empty, i.e., that it contains at least one object.

The domain of our example model $\mathfrak{M}$, written $|\mathfrak{M}|$, is {0, 1, 2}.

23.2.2 Interpretation function

- An *interpretation function* is an assignment of semantic value to each operation letter and predicate letter.

1 Chapter 4.3 on page 24

The interpretation function for model $\mathfrak{M}$ is $I_{\mathfrak{M}}$.

Operation letters

- To each **constant symbol** (i.e., zero-place operation letter) is assigned a member of the domain.

Intuitively, the constant symbol names the object, a member of the domain. If the domain is $|\mathfrak{M}|$ above and a_0^0 is assigned 0, then we think of a_0^0 naming 0 just as the name 'Socrates' names the man Socrates or the numeral '0' names the number 0. The assignment of 0 to a_0^0 can be expressed as:

$$I_{\mathfrak{M}}(a_0^0) = 0 \ .$$

- To each ***n*-place operation letter** with n greater than zero is assigned an $n{+}1$ place function taking ordered n-tuples of objects (members of the domain) as its arguments and objects (members of the domain) as its values. The function must be defined on all n-tuples of members of the domain.

Suppose the domain is $|\mathfrak{M}|$ above and we have a 2-place operation letter f_0^2. The function assigned to f_0^2 must then be defined on each ordered pair from the domain. For example, it can be the function $\mathfrak{f}_0^2$ such that:

$$\mathfrak{f}_0^2(0,0) = 2, \quad \mathfrak{f}_0^2(0,1) = 1, \quad \mathfrak{f}_0^2(0,2) = 0,$$

$$\mathfrak{f}_0^2(1,0) = 1, \quad \mathfrak{f}_0^2(1,1) = 0, \quad \mathfrak{f}_0^2(1,2) = 2,$$

$$\mathfrak{f}_0^2(2,0) = 0, \quad \mathfrak{f}_0^2(2,1) = 2, \quad \mathfrak{f}_0^2(2,2) = 1 \ .$$

The assignment to the operation letter is written as:

$$I_{\mathfrak{M}}(f_0^2) = \mathfrak{f}_0^2 \ .$$

Suppose that a_0^0 is assigned 0 as above and also that b_0^0 is assigned 1. Then we can intuitively think of the informally written $f(a,b)$ as naming (referring to, having the value) 1. This is analogous to 'the most famous student of Socrates' naming (or referring to) Plato or '2 + 3' naming (having the value) 5.

Predicate letters

- To each **sentence letter** (i.e., zero-place predicate letter) is assigned a truth value. For π a sentence letter, either

$$I_{\mathfrak{M}}(\pi) = \text{True}$$

or

$$I_{\mathfrak{M}}(\pi) = \text{False} \ .$$

This is the same treatment sentence letters received in sentential logic. Intuitively, the sentence is true or false accordingly as the sentence letter is assigned the value 'True' or 'False'.

- To each ***n*-place predicate letter** with n greater than zero is assigned an n-place relation (a set of ordered n-tuples) of members of the domain.

Intuitively, the predicate is true of each n-tuple in the assigned set. Let the domain be $|\mathfrak{M}|$ above and assume the assignment

$$I_{\mathfrak{M}}(\mathrm{F}^2_0) = \{<0,\ 1>,\ <1,\ 2>,\ <2,\ 1>\} \ .$$

Suppose that a^0_0 is assigned 0, b^0_0 is assigned 1, and c^0_0 is assigned 2. Then intuitively $\mathrm{F}(a,b)$, $\mathrm{F}(b,c)$, and $\mathrm{F}(c,b)$ should each be true. However, $\mathrm{F}(a,c)$, among others, should be false. This is analogous to 'is snub-nosed' being true of Socrates and 'is greater than' being true of <2, 3>.

23.2.3 Summary

The definition is interspersed with examples and so rather spread out. Here is a more compact summary. A model consists of two parts: a domain and interpretation function.

- A *domain* is a non-empty set.
- An *interpretation function* is an assignment of semantic value to each operation letter and predicate letter. This assignment proceeds as follows:
  - To each constant symbol (i.e., zero-place operation letter) is assigned a member of the domain.
  - To each n-place operation letter with n greater than zero is assigned an n+1 place function taking ordered n-tuples of objects (members of the domain) as its arguments and objects (members of the domain) as its values.
- To each sentence letter (i.e., zero-place predicate letter) is assigned a truth value.
- To each ***n*-place predicate letter** with n greater than zero is assigned an n-place relation (aset of ordered n-tuples) of members of the domain.

23.3 Examples

23.3.1 A finite model

An example model was specified in bits and pieces above. These pieces, collected together under the name $\mathfrak{M}$, are:

$|\mathfrak{M}| \;=\; \{1,\ 2,\ 3\}\ .$

$I_{\mathfrak{M}}(a_0^0) \;=\; 0\ .$

$I_{\mathfrak{M}}(b_0^0) \;=\; 1\ .$

$I_{\mathfrak{M}}(c_0^0) \;=\; 2\ .$

$I_{\mathfrak{M}}(f_0^2) \;=\; \mathfrak{f}_0^2$ such that: $\mathfrak{f}_0^2(0,0)=2,\ \mathfrak{f}_0^2(0,1) \;=\; 1,\ \mathfrak{f}_0^2(0,2)=0,$

$$\mathfrak{f}_0^2(1,0)=1,\ \mathfrak{f}_0^2(1,1) \;=\; 0,\ \mathfrak{f}_0^2(1,2)=2,\ \mathfrak{f}_0^2(2,0)=0,$$

$$\mathfrak{f}_0^2(2,1)=2,\ \text{and}\ \mathfrak{f}_0^2(2,2)=1\ .$$

$I_{\mathfrak{M}}(\mathrm{F}_0^2) \;=\; \{<0,\ 1>,\ <1,\ 2>,\ <2,\ 1>\}\ .$

We have not yet defined the rules for generating the semantic values of larger expressions. However, we can see some simple results we want that definition to achieve. A few such results have already been described:

$$f(a,b)\ \text{resolves to 1 in}\ \mathfrak{M}\ .$$

$$\mathrm{F}(a,b),\ \mathrm{F}(b,c),\ \text{and}\ \mathrm{F}(c,b)\ \text{are True in}\ \mathfrak{M}\ .$$

$$\mathrm{F}(a,a)\ \text{is False in}\ \mathfrak{M}\ .$$

Some more desired results can be added:

$$\mathrm{F}(a,f(a,b))\ \text{is True in}\ \mathfrak{M}\ .$$

$$\mathrm{F}(f(a,b),a)\ \text{is False in}\ \mathfrak{M}\ .$$

$$\mathrm{F}(c,b)\rightarrow\mathrm{F}(a,b)\ \text{is True in}\ \mathfrak{M}\ .$$

$$\mathrm{F}(c,b)\rightarrow\mathrm{F}(b,a)\ \text{is False in}\ \mathfrak{M}\ .$$

We can temporarily pretend that the numerals '0', '1', and '2' are added to $\mathcal{L}_{\mathcal{P}}$ and assign then the numbers 0, 1, and 2 respectively. We then want:

$$(1)\quad \mathrm{F}(0,1)\rightarrow\mathrm{F}(1,0)\quad \text{False in}\ \mathfrak{M}\ .$$

$$(2)\quad \mathrm{F}(1,2)\wedge\mathrm{F}(2,1)\quad \text{True in}\ \mathfrak{M}\ .$$

Because of (1), we will want as a result:

$$\forall x \forall y\,(\mathrm{F}(x,y) \rightarrow \mathrm{F}(y,x)) \text{ is false in } \mathfrak{M}\ .$$

Because of (2), we will want as a result:

$$\exists x\, \exists y\,(\mathrm{F}(x,y) \wedge \mathrm{F}(y,x)) \text{ is true in } \mathfrak{M}\ .$$

23.3.2 An infinite model

The domain $|\mathfrak{M}|$ had finitely many members; 3 to be exact. Models can have infinitely many members. Below is an example model $\mathfrak{M}_2$ with an infinitely large domain.

The domain $|\mathfrak{M}_2|$ is the set of natural numbers:

$$|\mathfrak{M}_2| = \{0, 1, 2, ...\}$$

The assignments to individual constant symbols can be as before:

$$I_{\mathfrak{M}_2}(a_0^0) \;=\; 0\ .$$

$$I_{\mathfrak{M}_2}(b_0^0) \;=\; 1\ .$$

$$I_{\mathfrak{M}_2}(c_0^0) \;=\; 2\ .$$

The 2-place operation letter f can be assigned, for example, the addition function:

$$I_{\mathfrak{M}_2}(f_0^2) \;=\; \mathfrak{f}_0^2 \text{ such that } \mathfrak{f}_0^2(u,v) = u+v\ .$$

The 2-place predicate letter F_0^2 can be assigned, for example, the *less than* relation:

$$I_{\mathfrak{M}_2}(\mathrm{F}_0^2) \;=\; \{< x,\ y >:\ x < y\}\ .$$

Some results that should be produced by the specification of an extended model:

$$f(a,b) \text{ resolves to } 1 \text{ in } \mathfrak{M}_2\ .$$

$$\mathrm{F}(a,b) \text{ and } \mathrm{F}(b,c) \text{ are True in } \mathfrak{M}_2\ .$$

$$\mathrm{F}(c,b) \text{ and } \mathrm{F}(a,a) \text{ are False in } \mathfrak{M}_2\ .$$

For every x, there is a y such that $x < y$. Thus we want as a result:

$$\forall x\, \exists y\, \mathrm{F}(x, y) \text{ is true in } \mathfrak{M}_2 \ .$$

There is no y such that $y < 0$ (remember, we are restricting ourselves to the domain which has no number less than 0). So it is not the case that, for every x, there is a y such that $y < x$. Thus we want as a result:

$$\forall x\, \exists y\, \mathrm{F}(y, x) \text{ is false in } \mathfrak{M}_2 \ .$$

24 Satisfaction

The rules for assigning truth to sentences of $\mathcal{L}_{\mathcal{P}}$ should say, in effect, that

$$\forall\alpha\,\varphi$$

is true if and only if φ is true of every object in the domain. There are two problems. First, φ will normally have free variables. In particular, it will normally have free α. But formulae with free variables are not sentences and do not have a truth value. Second, we do not yet have a precise way of saying that φ is true of every object in the domain. The solution to these problems comes in two parts.

- We will need an assignment of objects from the domain to the variables.
- We will need to say that a model satisfies (or does not satisfy) a formula with a variable assignment.

We can then define truth in a model in terms of satisfaction.

24.1 Variable assignment

Given model $\mathfrak{M}$, a *variable assignment* s is a function assigning each variable of $\mathcal{L}_{\mathcal{P}}$ a member of $|\mathfrak{M}|$. The function s is defined for all variables of $\mathcal{L}_{\mathcal{P}}$, so each one is assigned a member of the domain.

Okay, we have assignments of domain members to variables. We also have an assignment of domain members to constant symbols—this achieved by the model's interpretation function. Now we need to use this information to generate assignments of domain members to arbitrary terms including, in addition to constant symbols and variables, complex terms formed by using n-place operation letters where n is greater than 0. This is accomplished by an *extended variable assignment* $\bar{\mathrm{s}}$ defined below. Remember that $I_{\mathfrak{M}}$ is the interpretation function of model $\mathfrak{M}$. It assigns semantic values to the operation letters and predicate letters of $\mathcal{L}_{\mathcal{P}}$.

An extended variable assignment $\bar{\mathrm{s}}$ is a function making assignments as follows.

If α is a **variable**, then:

$$\bar{\mathrm{s}}(\alpha) \;=\; \mathrm{s}(\alpha)$$

If α is a **constant symbol** (i.e., a 0-place operation letter), then:

$$\bar{\mathrm{s}}(\alpha) \;=\; I_{\mathfrak{M}}(\alpha)$$

If ζ is an n-place operation letter (n greater than 0) and $\alpha_1,\ \alpha_2,\ \ldots,\ \alpha_n$ are **terms**, then:

$$\bar{\mathrm{s}}(\,\zeta(\alpha_1,\alpha_2,\ldots,\alpha_n)\,)\ =\ I_{\mathfrak{M}}(\zeta)\,(\ \bar{\mathrm{s}}(\alpha_1),\bar{\mathrm{s}}(\alpha_2),\ldots,\bar{\mathrm{s}}(\alpha_n)\ )$$

Some examples may help. Suppose we have model $I_{\mathfrak{M}}$ where:

$|\mathfrak{M}|\ =\ \{1,2,3\}$.

$I_{\mathfrak{M}}(a_0^0)\ =\ 0$.

$I_{\mathfrak{M}}(b_0^0)\ =\ 1$.

$I_{\mathfrak{M}}(f_0^2)\ =\ \mathfrak{f}_0^2$ such that: $\mathfrak{f}_0^2(0,0)=2,\ \mathfrak{f}_0^2(0,1)\ =\ 1,\ \mathfrak{f}_0^2(0,2)=0,$

$$\mathfrak{f}_0^2(1,0)=1,\ \mathfrak{f}_0^2(1,1)\ =\ 0,\ \mathfrak{f}_0^2(1,2)=2,\ \mathfrak{f}_0^2(2,0)=0,$$

$$\mathfrak{f}_0^2(2,1)=2,\ \text{and}\ \mathfrak{f}_0^2(2,2)=1\ .$$

On the previous page[1], it was noted that we want the following result:

$$f(a,b)\ \text{resolves to 1 in}\ \mathfrak{M}\ .$$

We now have achieved this because we have for any s defined on $I_{\mathfrak{M}}$:

$\bar{\mathrm{s}}(f(a,b))\ =\ I_{\mathfrak{M}}(f)\,(\ \bar{\mathrm{s}}(a),\bar{\mathrm{s}}(b)\ )\ =\ \mathfrak{f}_0^2(\ \bar{\mathrm{s}}(a),\bar{\mathrm{s}}(b)\ )$

$$=\ \mathfrak{f}_0^2(\ I_{\mathfrak{M}}(a),I_{\mathfrak{M}}(b)\ )\ =\ \mathfrak{f}_0^2(0,1)\ =\ 1\ .$$

Suppose we also have a variable assignment s where:

$$\mathrm{s}(x_0)\ =\ 0\ .$$

$$\mathrm{s}(y_0)\ =\ 1\ .$$

Then we get:

$\bar{\mathrm{s}}(f(x,y))\ =\ I_{\mathfrak{M}}(f)\,(\ \bar{\mathrm{s}}(x),\bar{\mathrm{s}}(y)\ )\ =\ \mathfrak{f}_0^2(\ \bar{\mathrm{s}}(x),\bar{\mathrm{s}}(y)\ )$

$$=\ \mathfrak{f}_0^2(f)\,(\ \mathrm{s}(x),\mathrm{s}(y)\ )\ =\ \mathfrak{f}_0^2(0,1)\ =\ 1\ .$$

1 Chapter 22.4 on page 218

24.2 Satisfaction

A model, together with a variable assignment, can *satisfy* (or fail to satisfy) a formula. Then we will use the notion of satisfaction with a variable assignment to define truth of a sentence in a model. We can use the following convenient notation to say that the interpretaion $\mathfrak{M}$ satisfies (or does not satisfy) φ with s.

$$< \mathfrak{M},\ \mathrm{s} > \vDash \varphi$$

$$< \mathfrak{M},\ \mathrm{s} > \nvDash \varphi$$

We now define *satisfaction of a formula by a model with a variable assignment.* In the following, 'iff' is used to mean 'if and only if'.

i. For σ a sentence letter:

$$< \mathfrak{M},\ \mathrm{s} > \vDash \sigma \quad \text{iff} \quad I_{\mathfrak{M}}(\sigma) \ = \ \text{True} \ .$$

ii. For π an n–place predicate letter and for $\alpha_0, \alpha_1, \ldots, \alpha_n$ any terms:

$$< \mathfrak{M},\ \mathrm{s} > \vDash \pi(\alpha_0, \alpha_1, \ldots, \alpha_n) \quad \text{iff} \ < \bar{\mathrm{s}}(\alpha_0), \bar{\mathrm{s}}(\alpha_1), \ldots, \bar{\mathrm{s}}(\alpha_n) > \in \ I_{\mathfrak{M}}(\pi) \ .$$

iii. For φ a formula:

$$< \mathfrak{M},\ \mathrm{s} > \vDash \neg\varphi \quad \text{iff} \ < \mathfrak{M},\ \mathrm{s} > \nvDash \varphi \ .$$

iv. For φ and ψ formulae:

$$< \mathfrak{M},\ \mathrm{s} > \vDash (\varphi \wedge \psi) \quad \text{iff} \ < \mathfrak{M},\ \mathrm{s} > \vDash \varphi \ \text{ and } \ < \mathfrak{M},\ \mathrm{s} > \vDash \psi \ .$$

v. For φ and ψ formulae:

$$< \mathfrak{M},\ \mathrm{s} > \vDash (\varphi \vee \psi) \quad \text{iff} \ < \mathfrak{M},\ \mathrm{s} > \vDash \varphi \ \text{ or } \ < \mathfrak{M},\ \mathrm{s} > \vDash \psi \ \text{(or both)} \ .$$

vi. For φ and ψ formulae:

$$< \mathfrak{M},\ \mathrm{s} > \vDash (\varphi \rightarrow \psi) \quad \text{iff} \ < \mathfrak{M},\ \mathrm{s} > \nvDash \varphi \ \text{ or } \ < \mathfrak{M},\ \mathrm{s} > \vDash \psi \ \text{(or both)} \ .$$

vii. For φ and ψ formulae:

$$< \mathfrak{M},\ \mathrm{s} > \vDash (\varphi \leftrightarrow \psi) \quad \text{iff} \ < \mathfrak{M},\ \mathrm{s} > \vDash (\varphi \wedge \psi) \ \text{ or } \ < \mathfrak{M},\ \mathrm{s} > \vDash (\neg\varphi \wedge \neg\psi) \ \text{(or both)} \ .$$

viii. For φ a formula and α a variable:

$$< \mathfrak{M},\ \mathrm{s} > \vDash \forall\alpha\varphi \quad \text{iff for every d } \in |\mathfrak{M}|, \ < \mathfrak{M},\ \mathrm{s}[\alpha : \mathrm{d}] > \vDash \varphi \ .$$

where s$[\alpha : \mathrm{d}]$ differs from s only in assigning d to α .

ix. For φ a formula and α a variable:

$$< \mathfrak{M},\ \mathrm{s} > \vDash \exists\alpha\varphi \quad \text{iff} \ \text{ for at least one d} \in |\mathfrak{M}|,\ < \mathfrak{M},\ \mathrm{s}[\alpha : \mathrm{d}] > \vDash \varphi\ .$$

where s$[\alpha : \mathrm{d}]$ differs from s only in assigning d to α .

24.3 Examples

The following continue the examples used when describing extended variable assignments above. They are based on the examples of the previous page[2].

24.3.1 A model and variable assignment for examples

Suppose we have model $I_{\mathfrak{M}}$ where

$|\mathfrak{M}| \ = \ \{0, 1, 2\}$.

$I_{\mathfrak{M}}(a_0^0) \ = \ 0$.

$I_{\mathfrak{M}}(b_0^0) \ = \ 1$.

$I_{\mathfrak{M}}(c_0^0) \ = \ 2$.

$I_{\mathfrak{M}}(f_0^2) \ = \ \mathfrak{f}_0^2$ such that: $\mathfrak{f}_0^2(0,0) = 2,\ \mathfrak{f}_0^2(0,1) \ = \ 1,\ \mathfrak{f}_0^2(0,2) = 0,$

$$\mathfrak{f}_0^2(1,0) = 1,\ \mathfrak{f}_0^2(1,1) \ = \ 0,\ \mathfrak{f}_0^2(1,2) = 2,\ \mathfrak{f}_0^2(2,0) = 0,$$

$$\mathfrak{f}_0^2(2,1) = 2,\ \text{and}\ \mathfrak{f}_0^2(2,2) = 1\ .$$

$I_{\mathfrak{M}}(\mathrm{F}_0^2) \ = \ \{< 0,\ 1 >,\ < 1,\ 2 >,\ < 2,\ 1 >\}$.

Suppose further we have a variable assignment s where:

$$\mathrm{s}(x_0) \ = \ 0\ .$$

$$\mathrm{s}(y_0) \ = \ 1\ .$$

$$\mathrm{s}(z_0) \ = \ 2\ .$$

2 Chapter 22.4 on page 218

We already saw that both of the following resove to 1:

$$f(a,b)$$

$$f(x,y)$$

24.3.2 Examples without quantifiers

Given model $\mathfrak{M}$, the previous page[3] noted the following further goals:

$$(1) \quad \mathrm{F}(a,b),\ \mathrm{F}(b,c),\ \text{and } \mathrm{F}(c,b) \text{ are True in } \mathfrak{M}\ .$$

$$(2) \quad \mathrm{F}(a,a) \text{ is False in } \mathfrak{M}\ .$$

$$(3) \quad \mathrm{F}(a,f(a,b)) \text{ is True in } \mathfrak{M}\ .$$

$$(4) \quad \mathrm{F}(f(a,b),a) \text{ is False in } \mathfrak{M}\ .$$

$$(5) \quad \mathrm{F}(c,b) \to \mathrm{F}(a,b) \text{ is True in } \mathfrak{M}\ .$$

$$(6) \quad \mathrm{F}(c,b) \to \mathrm{F}(b,a) \text{ is False in } \mathfrak{M}\ .$$

We are not yet ready to evaluate for truth or falsity, but we can take a step in that direction by seeing that these sentences are satisfied by $\mathfrak{M}$ with s. Indeed, the details of s will not figure in dertermining which of these are satisfied. Thus $\mathfrak{M}$ satisfies (or fails to satisfy) them with any variable assignment. As we will see on the next page, that is the criterion for truth (or falsity) in $\mathfrak{M}$.

Corresponding to (1),

$$< \mathfrak{M},\ \mathrm{s} > \vDash \mathrm{F}(a,b),\ \mathrm{F}(b,c),\ \text{and } \mathrm{F}(c,b)\ .$$

In particular:

$$< \mathfrak{M},\ \mathrm{s} > \vDash \mathrm{F}(a,b) \quad \text{because} \quad < \bar{\mathrm{s}}(a), \bar{\mathrm{s}}(b) > \; = \; < 0, 1 > \in \ I_{\mathfrak{M}}(\mathrm{F})\ .$$

$$< \mathfrak{M},\ \mathrm{s} > \vDash \mathrm{F}(b,c) \quad \text{because} \quad < \bar{\mathrm{s}}(b), \bar{\mathrm{s}}(c) > \; = \; < 1, 2 > \in \ I_{\mathfrak{M}}(\mathrm{F})\ .$$

3 Chapter 22.4 on page 218

$$<\mathfrak{M},\ \mathrm{s}> \vDash \mathrm{F}(c,b) \ \text{ because } <\bar{\mathrm{s}}(c), \bar{\mathrm{s}}(b)> \,=\, <2,1> \,\in\, I_{\mathfrak{M}}(\mathrm{F})\ .$$

Corresponding to (2) through (6) respectively:

$$<\mathfrak{M},\ \mathrm{s}> \nvDash \mathrm{F}(a,a)) \ \text{ because } <\bar{\mathrm{s}}(a), \bar{\mathrm{s}}(a)> \,=\, <0,0> \,\notin\, I_{\mathfrak{M}}(\mathrm{F})\ .$$

$$<\mathfrak{M},\ \mathrm{s}> \vDash \mathrm{F}(a,f(a,b)) \ \text{ because } <\bar{\mathrm{s}}(a), \bar{\mathrm{s}}(f(a,b))> \,=\, <0,1> \,\in\, I_{\mathfrak{M}}(\mathrm{F})\ .$$

$$<\mathfrak{M},\ \mathrm{s}> \nvDash \mathrm{F}(f(a,b),a) \ \text{ because } <\bar{\mathrm{s}}(f(a,b), \bar{\mathrm{s}}(a)> \,=\, <1,0> \,\notin\, I_{\mathfrak{M}}(\mathrm{F})\ .$$

$$<\mathfrak{M},\ \mathrm{s}> \vDash \mathrm{F}(c,b) \to \mathrm{F}(a,b) \ \text{ because } <\mathfrak{M},\ \mathrm{s}> \vDash \mathrm{F}(a,b)\ .$$

$$<\mathfrak{M},\ \mathrm{s}> \nvDash \mathrm{F}(c,b) \to \mathrm{F}(b,a) \ \text{ because } <\mathfrak{M},\ \mathrm{s}> \vDash \mathrm{F}(c,b) \text{ but } <\mathfrak{M},\ \mathrm{s}> \nvDash \mathrm{F}(b,a)\ .$$

As noted above, the details of s were not relevant to these evaluations. But for similar formulae using free variables instead of constant symbols, the details or s do become relevant. Examples based the above are:

$$<\mathfrak{M},\ \mathrm{s}> \vDash \mathrm{F}(x,y) \ \text{ because } <\bar{\mathrm{s}}(x), \bar{\mathrm{s}}(y)> \,=\, <0,1> \,\in\, I_{\mathfrak{M}}(\mathrm{F})\ .$$

$$<\mathfrak{M},\ \mathrm{s}> \vDash \mathrm{F}(y,z) \ \text{ because } <\bar{\mathrm{s}}(y), \bar{\mathrm{s}}(z)> \,=\, <1,2> \,\in\, I_{\mathfrak{M}}(\mathrm{F})\ .$$

$$<\mathfrak{M},\ \mathrm{s}> \vDash \mathrm{F}(z,y) \ \text{ because } <\bar{\mathrm{s}}(z), \bar{\mathrm{s}}(y)> \,=\, <2,1> \,\in\, I_{\mathfrak{M}}(\mathrm{F})\ .$$

$$<\mathfrak{M},\ \mathrm{s}> \nvDash \mathrm{F}(x,x)) \ \text{ because } <\bar{\mathrm{s}}(x), \bar{\mathrm{s}}(x)> \,=\, <0,0> \,\notin\, I_{\mathfrak{M}}(\mathrm{F})\ .$$

$$<\mathfrak{M},\ \mathrm{s}> \vDash \mathrm{F}(x,f(x,y)) \ \text{ because } <\bar{\mathrm{s}}(x), \bar{\mathrm{s}}(f(x,y))> \,=\, <0,1> \,\in\, I_{\mathfrak{M}}(\mathrm{F})\ .$$

$$<\mathfrak{M},\ \mathrm{s}> \nvDash \mathrm{F}(f(x,y),x) \ \text{ because } <\bar{\mathrm{s}}(f(x,y), \bar{\mathrm{s}}(x)> \,=\, <1,0> \,\notin\, I_{\mathfrak{M}}(\mathrm{F})\ .$$

$$<\mathfrak{M},\ \mathrm{s}> \vDash \mathrm{F}(z,y) \to \mathrm{F}(x,y) \ \text{ because } <\mathfrak{M},\ \mathrm{s}> \vDash \mathrm{F}(x,y)\ .$$

$$<\mathfrak{M},\ \mathrm{s}> \nvDash \mathrm{F}(z,y) \to \mathrm{F}(y,x) \ \text{ because } <\mathfrak{M},\ \mathrm{s}> \vDash \mathrm{F}(z,y) \text{ but } <\mathfrak{M},\ \mathrm{s}> \nvDash \mathrm{F}(y,x)\ .$$

24.3.3 Examples with quantifiers

Given model $\mathfrak{M}$, the previous page[4] also noted the following further goals:

$$(7)\quad \forall x \forall y\,(\mathrm{F}(x,y) \rightarrow \mathrm{F}(y,x)) \text{ is false in } \mathfrak{M}\,.$$

$$(8)\quad \exists x\,\exists y\,(\mathrm{F}(x,y) \wedge \mathrm{F}(y,x)) \text{ is true in } \mathfrak{M}\,.$$

Again, we are not yet ready to evaluate for truth or falsity, but again we can take a step in that direction by seeing that the sentence in (7) is and the sentence in (8) is not satisfied by $\mathfrak{M}$ with s.

Corresponding to (7):

$$(9)\quad <\mathfrak{M},\ \mathrm{s}> \nvDash \forall x \forall y\,(\mathrm{F}(x,y) \rightarrow \mathrm{F}(y,x))$$

is true if and only if at least one of the following is true:

$$(10)\quad <\mathfrak{M},\ \mathrm{s}[x:0]> \nvDash \forall y\,(\mathrm{F}(x,y) \rightarrow \mathrm{F}(y,x))$$

$$(11)\quad <\mathfrak{M},\ \mathrm{s}[x:1]> \nvDash \forall y\,(\mathrm{F}(x,y) \rightarrow \mathrm{F}(y,x))$$

$$(12)\quad <\mathfrak{M},\ \mathrm{s}[x:2]> \nvDash \forall y\,(\mathrm{F}(x,y) \rightarrow \mathrm{F}(y,x))$$

The formula of (7) and (9) is satisfied by $<\mathfrak{M},\ \mathrm{s}>$ if and only if it is satified by $\mathfrak{M}$ with each of the modified variable assignments. Turn this around, and we get the formula failing to be satisfied by $<\mathfrak{M},\ \mathrm{s}>$ if and only if it fails to be satisfied by the model with at least one of the three modified variable assignments as per (10) through (12). Similarly, (10) is true if and only if at least one of the following are true:

$$<\mathfrak{M},\ \mathrm{s}[x:0,\ y:0]> \nvDash \mathrm{F}(x,y) \rightarrow \mathrm{F}(y,x)$$

$$<\mathfrak{M},\ \mathrm{s}[x:0,\ y:1]> \nvDash \mathrm{F}(x,y) \rightarrow \mathrm{F}(y,x)$$

$$<\mathfrak{M},\ \mathrm{s}[x:0,\ y:2]> \nvDash \mathrm{F}(x,y) \rightarrow \mathrm{F}(y,x)$$

Indeed, the middle one of these is true. This is because

$$<0,1> \in\ I_{\mathfrak{M}}(\mathrm{F}) \text{ but } <1,0> \notin\ I_{\mathfrak{M}}(\mathrm{F})\ .$$

4 Chapter 22.4 on page 218

Thus (9) is true.

Corresponding to (8),

$$(13) \quad <\mathfrak{M},\ \mathrm{s}> \vDash \exists x\, \exists y\, (\mathrm{F}(x,y) \wedge \mathrm{F}(y,x))$$

is true if and only if at least one of the following is true:

$$<\mathfrak{M},\ \mathrm{s}[x:0]> \vDash \exists y\, (\mathrm{F}(x,y) \wedge \mathrm{F}(y,x))$$

$$<\mathfrak{M},\ \mathrm{s}[x:1]> \vDash \exists y\, (\mathrm{F}(x,y) \wedge \mathrm{F}(y,x))$$

$$<\mathfrak{M},\ \mathrm{s}[x:2]> \vDash \exists y\, (\mathrm{F}(x,y) \wedge \mathrm{F}(y,x))$$

The middle of these is true if and only if at least one of the following are true:

$$<\mathfrak{M},\ \mathrm{s}[x:1,\ y:0]> \vDash \mathrm{F}(x,y) \wedge \mathrm{F}(y,x)$$

$$<\mathfrak{M},\ \mathrm{s}[x:1,\ y:1]> \vDash \mathrm{F}(x,y) \wedge \mathrm{F}(y,x)$$

$$<\mathfrak{M},\ \mathrm{s}[x:1,\ y:2]> \vDash \mathrm{F}(x,y) \wedge \mathrm{F}(y,x)$$

Indeed, the last of these is true. This is because:

$$<1,2> \in\ I_{\mathfrak{M}}(\mathrm{F}) \text{ and } <2,1> \in\ I_{\mathfrak{M}}(\mathrm{F})\ .$$

Thus (13) is true.

25 Truth

25.1 Truth in a model

We have defined satisfaction in a model with a variable assignment. We have expressed formula φ being satisfied by model $\mathfrak{M}$ with variable assignment s as:

$$< \mathfrak{M},\ s > \vDash \varphi$$

Now we can also say that a formula φ is satisfied by model $\mathfrak{M}$ (not limited to a specific variable assignment) if and only if φ is satisfied by $\mathfrak{M}$ with every variable assignment. Thus

$$\mathfrak{M} \vDash \varphi$$

if and only if

$$< \mathfrak{M},\ s > \vDash \varphi \text{ for every s .}$$

If no free variables occur in φ, (that is, if φ is a sentence), then φ is true in model $\mathfrak{M}$.

Variable assignments allow us to deal with free variables when doing the semantic analysis of a formula. For two variable assignments, s_1 and s_2, satisfaction by $< \mathfrak{M},\ s_1 >$ differs from satisfaction by $< \mathfrak{M},\ s_2 >$ only if the formula has free variables. But sentences do not have free variables. Thus a model satisfies a sentence with at least one variable assignment if and only if it satisfies the sentence with every variable assignment. The following two definitions are equivalent:

φ is true in $\mathfrak{M}$ if and only if there is a variable assignment s such that

$$< \mathfrak{M},\ s > \vDash \varphi\ .$$

φ is true in $\mathfrak{M}$ if and only if, for every variable assignment s,

$$< \mathfrak{M},\ s > \vDash \varphi\ .$$

The latter is just a notational variant of:

φ is true in $\mathfrak{M}$ if and only if

$$\mathfrak{M} \vDash \varphi\ .$$

25.2 Examples

25.2.1 A finite model

The example model

On the previous page[1], we looked at the following model and variable assignment.

For the model $I_{\mathfrak{M}}$:

$|\mathfrak{M}| \ = \ \{1, 2, 3\}$.

$I_{\mathfrak{M}}(a_0^0) \ = \ 0$.

$I_{\mathfrak{M}}(b_0^0) \ = \ 1$.

$I_{\mathfrak{M}}(c_0^0) \ = \ 2$.

$I_{\mathfrak{M}}(f_0^2) \ = \ \mathfrak{f}_0^2$ such that: $\mathfrak{f}_0^2(0,0) = 2,\ \mathfrak{f}_0^2(0,1) \ = \ 1,\ \mathfrak{f}_0^2(0,2) = 0,$

$$\mathfrak{f}_0^2(1,0) = 1,\ \mathfrak{f}_0^2(1,1) \ = \ 0,\ \mathfrak{f}_0^2(1,2) = 2,\ \mathfrak{f}_0^2(2,0) = 0,$$

$$\mathfrak{f}_0^2(2,1) = 2,\ \text{and}\ \mathfrak{f}_0^2(2,2) = 1\ .$$

$I_{\mathfrak{M}}(\mathrm{F}_0^2) \ = \ \{< 0,\ 1 >,\ < 1,\ 2 >,\ < 2,\ 1 >\}$.

For the variable assignment s :

$$\mathrm{s}(x_0) \ = \ 0\ .$$

$$\mathrm{s}(y_0) \ = \ 1\ .$$

$$\mathrm{s}(z_0) \ = \ 2\ .$$

Example results

We noted the following results:

$$< \mathfrak{M},\ \mathrm{s} > \vDash \mathrm{F}(a,b),\ \mathrm{F}(b,c),\ \text{and}\ \mathrm{F}(c,b)\ .$$

$$< \mathfrak{M},\ \mathrm{s} > \nvDash \mathrm{F}(a,a))\ .$$

1 Chapter 23.3.2 on page 224

$$< \mathfrak{M},\ \mathrm{s} > \vDash \mathrm{F}(a, f(a,b))\ .$$

$$< \mathfrak{M},\ \mathrm{s} > \nvDash \mathrm{F}(f(a,b),a)\ .$$

$$< \mathfrak{M},\ \mathrm{s} > \vDash \mathrm{F}(c,b) \to \mathrm{F}(a,b)\ .$$

$$< \mathfrak{M},\ \mathrm{s} > \nvDash \mathrm{F}(c,b) \to \mathrm{F}(b,a)\ .$$

$$< \mathfrak{M},\ \mathrm{s} > \nvDash \forall x \forall y\, (\mathrm{F}(x,y) \to \mathrm{F}(y,x))$$

$$< \mathfrak{M},\ \mathrm{s} > \vDash \exists x\, \exists y\, (\mathrm{F}(x,y) \wedge \mathrm{F}(y,x))$$

We also noted above that for sentences (though not for formulae in general), a model satisfies the sentence with at least one variable assignment if and only if it satisfies the sentence with every variable assignment. Thus the results just listed hold for every variable assignment, not just s.

Applying our definition of truth, we get:

$$(1)\quad \mathrm{F}(a,b),\ \mathrm{F}(b,c),\ \text{and } \mathrm{F}(c,b) \text{ are True in } \mathfrak{M}\ .$$

$$(2)\quad \mathrm{F}(a,a) \text{ is False in } \mathfrak{M}\ .$$

$$(3)\quad \mathrm{F}(a, f(a,b)) \text{ is True in } \mathfrak{M}\ .$$

$$(4)\quad \mathrm{F}(f(a,b),a) \text{ is False in } \mathfrak{M}\ .$$

$$(5)\quad \mathrm{F}(c,b) \to \mathrm{F}(a,b) \text{ is True in } \mathfrak{M}\ .$$

$$(6)\quad \mathrm{F}(c,b) \to \mathrm{F}(b,a) \text{ is False in } \mathfrak{M}\ .$$

$$(7)\quad \forall x \forall y\, (\mathrm{F}(x,y) \to \mathrm{F}(y,x)) \text{ is false in } \mathfrak{M}\ .$$

$$(8)\quad \exists x\, \exists y\, (\mathrm{F}(x,y) \wedge \mathrm{F}(y,x)) \text{ is true in } \mathfrak{M}\ .$$

This corresponds to the goals (1)−(8) of the previous page[2]. We have now achieved those goals.

25.2.2 An infinite model

The example model

On Models page[3], we also considered an infinite model $I_{\mathfrak{M}_2}$:

$$|\mathfrak{M}_2| = \{0,1,2,...\}$$

$$I_{\mathfrak{M}_2}(a_0^0) \ = \ 0 \ .$$

$$I_{\mathfrak{M}_2}(b_0^0) \ = \ 1 \ .$$

$$I_{\mathfrak{M}_2}(c_0^0) \ = \ 2 \ .$$

$$I_{\mathfrak{M}_2}(f_0^2) \ = \ \mathfrak{f}_0^2 \text{ such that } \mathfrak{f}_0^2(u,v) = u+v \ .$$

$$I_{\mathfrak{M}_2}(\mathrm{F}_0^2) \ = \ \{< x, \ y >: \ x < y\} \ .$$

We can reuse the same variable assignment from above, namely s :

$$\mathrm{s}(x_0) \ = \ 0 \ .$$

$$\mathrm{s}(y_0) \ = \ 1 \ .$$

$$\mathrm{s}(z_0) \ = \ 2 \ .$$

2 Chapter 23.3.2 on page 224
3 Chapter 22.4 on page 218

Example of extended varable assignment

On the Models page[4], we listed the following goals for our definitions.

$$f(a,b) \text{ resolves to 1 in } \mathfrak{M}_2 \ .$$

This does not require our definition of truth or the definition of satisfaction; it is simply requires evaluating the exended variable assignment. We have for any s on defined on $I_{\mathfrak{M}}$:

$$\bar{\mathrm{s}}(f(a,b)) \ = \ I_{\mathfrak{M}}(f)(\ \bar{\mathrm{s}}(a), \bar{\mathrm{s}}(b)\ ) \ = \ \mathfrak{f}_0^2(\ \bar{\mathrm{s}}(a), \bar{\mathrm{s}}(b)\ )$$

$$= \ \mathfrak{f}_0^2(\ I_{\mathfrak{M}}(a), I_{\mathfrak{M}}(b)\ ) \ = \ \mathfrak{f}_0^2(0,1)$$

$$= \ 0+1 \ = \ 1 \ .$$

Example results without quantifiers

We also listed the following goal on the Models page[5].

$$(9) \quad \mathrm{F}(a,b) \text{ and } \mathrm{F}(b,c) \text{ are True in } \mathfrak{M}_2 \ .$$

$$(10) \quad \mathrm{F}(c,b) \text{ and } \mathrm{F}(a,a) \text{ are False in } \mathfrak{M}_2 \ .$$

First we note that:

$$< \mathfrak{M}, \ \mathrm{s} > \vDash \mathrm{F}(a,b) \ \text{ because } \ < \bar{\mathrm{s}}(a), \bar{\mathrm{s}}(b) > \ = \ < 0, 1 > \ \in \ I_{\mathfrak{M}}(\mathrm{F})$$

$$< \mathfrak{M}, \ \mathrm{s} > \vDash \mathrm{F}(b,c) \ \text{ because } \ < \bar{\mathrm{s}}(b), \bar{\mathrm{s}}(c) > \ = \ < 1, 2 > \ \in \ I_{\mathfrak{M}}(\mathrm{F})$$

$$< \mathfrak{M}, \ \mathrm{s} > \nvDash \mathrm{F}(c,b) \ \text{ because } \ < \bar{\mathrm{s}}(c), \bar{\mathrm{s}}(b) > \ = \ < 2, 1 > \ \notin \ I_{\mathfrak{M}}(\mathrm{F})$$

$$< \mathfrak{M}, \ \mathrm{s} > \nvDash \mathrm{F}(a,a) \ \text{ because } \ < \bar{\mathrm{s}}(a), \bar{\mathrm{s}}(a) > \ = \ < 0, 0 > \ \notin \ I_{\mathfrak{M}}(\mathrm{F}) \ .$$

Indeed:

$$< 0, 1 > \ \in \ I_{\mathfrak{M}}(\mathrm{F}) \ \text{ because } \ 0 < 1$$

4 Chapter 22.4 on page 218
5 Chapter 22.4 on page 218

$$< 1, 2 > \in \ I_{\mathfrak{M}}(\mathrm{F}) \quad \text{because} \quad 1 < 2$$

$$< 2, 1 > \notin \ I_{\mathfrak{M}}(\mathrm{F}) \quad \text{because} \quad 2 \not< 1$$

$$< 0, 0 > \notin \ I_{\mathfrak{M}}(\mathrm{F}) \quad \text{because} \quad 0 \not< 0 \ .$$

Because the formulae of (9) and (10) are sentences,

$$< \mathfrak{M} > \vDash \mathrm{F}(a,b) \ .$$

$$< \mathfrak{M} > \vDash \mathrm{F}(b,c) \ .$$

$$< \mathfrak{M} > \nvDash \mathrm{F}(c,b) \ .$$

$$< \mathfrak{M} > \nvDash \mathrm{F}(a,a) \ .$$

Applying the definition of truth, we find the goals of (9) and (10) achieved. The sentences of (9) are true and those of (10) are false.

Example results with quantifiers

In addition, we listed the following goal on the Models page[6].

$$(11) \quad \forall x \, \exists y \, \mathrm{F}(x,y) \text{ is true in } \mathfrak{M}_2 \ .$$

$$(12) \quad \forall x \, \exists y \, \mathrm{F}(y,x) \text{ is false in } \mathfrak{M}_2 \ .$$

Corresponding to (11):

$$(13) \quad < \mathfrak{M}_2, \ \mathrm{s} > \vDash \forall x \, \exists y \, \mathrm{F}(x,y)$$

is true if and only if, for each i a member of the domain, the following is true of at least one j a member of the domain:

$$< \mathfrak{M}_2, \ \mathrm{s}[x : i, x : j] > \vDash \mathrm{F}(x,y) \ .$$

6 Chapter 22.4 on page 218

But F_0^2 was assigned the *less then* relation. Thus the preceding holds if and only if, for every member of the domain, there is a larger member of the domain. Given that the domain is $\{0, 1, 2, ...\}$, this is obviously true. Thus, (13) is true. Given that the formula of (11) and (12) is a sentence, we find the goal expressed as (11) to be met.

Corresponding to (12):

$$(14) \quad < \mathfrak{M}_2,\ \mathrm{s} > \vDash \forall x \exists y \, \mathrm{F}(y, x)$$

is true if and only if, for each i a member of the domain, the following is true of at least one j a member of the domain:

$$< \mathfrak{M}_2,\ \mathrm{s}[x : i, x : j] > \vDash \mathrm{F}(y, x) \ .$$

This holds if and only if, for every member of the domain, there is a smaller member of the domain. But there is no member of the domain smaller than 0. Thus (14) is false. The formula of (12) and (14) fails to be satisfied by $\mathfrak{M}_2$ with variable assignment s. The formula of (12) and (14) is a sentence, so it fails to be satisfied by $\mathfrak{M}_2$ with *any* variable assignment. The formula (a sentence) of (12) and (14) is false, and so the goal of (12) is met.

26 Contributors

Edits	User
1	Adrignola[1]
1	Aldoise[2]
1	AlgebraicSpore~enwikibooks[3]
1	Alsocal[4]
2	Anhtrobote[5]
13	Avicennasis[6]
2	BiT[7]
1	Cspurrier[8]
3	DialaceStarvy[9]
7	Dirk Hünniger[10]
2	Heinrich Zweihänder[11]
1	Herbythyme[12]
1	JECompton~enwikibooks[13]
609	JMRyan[14]
1	Jeremy.boyd[15]
1	Jguk[16]
1	Jomegat[17]
9	Jonathan m price[18]
2	Jwinder47[19]
2	Knewter[20]

1 https://en.wikibooks.org/wiki/User:Adrignola
2 https://en.wikibooks.org/w/index.php%3ftitle=User:Aldoise&action=edit&redlink=1
3 https://en.wikibooks.org/w/index.php%3ftitle=User:AlgebraicSpore~enwikibooks&action=edit&redlink=1
4 https://en.wikibooks.org/wiki/User:Alsocal
5 https://en.wikibooks.org/w/index.php%3ftitle=User:Anhtrobote&action=edit&redlink=1
6 https://en.wikibooks.org/wiki/User:Avicennasis
7 https://en.wikibooks.org/wiki/User:BiT
8 https://en.wikibooks.org/wiki/User:Cspurrier
9 https://en.wikibooks.org/w/index.php%3ftitle=User:DialaceStarvy&action=edit&redlink=1
10 https://en.wikibooks.org/wiki/User:Dirk_H%25C3%25BCnniger
11 https://en.wikibooks.org/w/index.php%3ftitle=User:Heinrich_Zweih%25C3%25A4nder&action=edit&redlink=1
12 https://en.wikibooks.org/wiki/User:Herbythyme
13 https://en.wikibooks.org/wiki/User:JECompton~enwikibooks
14 https://en.wikibooks.org/wiki/User:JMRyan
15 https://en.wikibooks.org/w/index.php%3ftitle=User:Jeremy.boyd&action=edit&redlink=1
16 https://en.wikibooks.org/wiki/User:Jguk
17 https://en.wikibooks.org/wiki/User:Jomegat
18 https://en.wikibooks.org/w/index.php%3ftitle=User:Jonathan_m_price&action=edit&redlink=1
19 https://en.wikibooks.org/w/index.php%3ftitle=User:Jwinder47&action=edit&redlink=1
20 https://en.wikibooks.org/w/index.php%3ftitle=User:Knewter&action=edit&redlink=1

1 Kowey[21]
1 Lewis Goudy[22]
1 Mauvarca~enwikibooks[23]
1 Moonty~enwikibooks[24]
6 Noamaster88[25]
5 Pbrower2a[26]
8 Peter Cross[27]
19 Publunch~enwikibooks[28]
1 QuiteUnusual[29]
9 Rnddim[30]
1 Russell208[31]
2 Sigma 7[32]
1 Sirbailey[33]
1 Spongebob88[34]
1 StephenBank[35]
1 The8thbit[36]
1 TorosFanny[37]
2 Torzsmokus[38]
1 Vampirolog~enwikibooks[39]
7 Van der Hoorn[40]
1 Vubb[41]
2 Zeiimer[42]
1 Zhaladshar[43]

21 https://en.wikibooks.org/wiki/User:Kowey
22 https://en.wikibooks.org/w/index.php%3ftitle=User:Lewis_Goudy&action=edit&redlink=1
23 https://en.wikibooks.org/w/index.php%3ftitle=User:Mauvarca~enwikibooks&action=edit&redlink=1
24 https://en.wikibooks.org/w/index.php%3ftitle=User:Moonty~enwikibooks&action=edit&redlink=1
25 https://en.wikibooks.org/w/index.php%3ftitle=User:Noamaster88&action=edit&redlink=1
26 https://en.wikibooks.org/w/index.php%3ftitle=User:Pbrower2a&action=edit&redlink=1
27 https://en.wikibooks.org/w/index.php%3ftitle=User:Peter_Cross&action=edit&redlink=1
28 https://en.wikibooks.org/wiki/User:Publunch~enwikibooks
29 https://en.wikibooks.org/wiki/User:QuiteUnusual
30 https://en.wikibooks.org/wiki/User:Rnddim
31 https://en.wikibooks.org/w/index.php%3ftitle=User:Russell208&action=edit&redlink=1
32 https://en.wikibooks.org/wiki/User:Sigma_7
33 https://en.wikibooks.org/w/index.php%3ftitle=User:Sirbailey&action=edit&redlink=1
34 https://en.wikibooks.org/w/index.php%3ftitle=User:Spongebob88&action=edit&redlink=1
35 https://en.wikibooks.org/w/index.php%3ftitle=User:StephenBank&action=edit&redlink=1
36 https://en.wikibooks.org/w/index.php%3ftitle=User:The8thbit&action=edit&redlink=1
37 https://en.wikibooks.org/w/index.php%3ftitle=User:TorosFanny&action=edit&redlink=1
38 https://en.wikibooks.org/w/index.php%3ftitle=User:Torzsmokus&action=edit&redlink=1
39 https://en.wikibooks.org/w/index.php%3ftitle=User:Vampirolog~enwikibooks&action=edit&redlink=1
40 https://en.wikibooks.org/wiki/User:Van_der_Hoorn
41 https://en.wikibooks.org/w/index.php%3ftitle=User:Vubb&action=edit&redlink=1
42 https://en.wikibooks.org/w/index.php%3ftitle=User:Zeiimer&action=edit&redlink=1
43 https://en.wikibooks.org/wiki/User:Zhaladshar

List of Figures

- GFDL: Gnu Free Documentation License. http://www.gnu.org/licenses/fdl.html
- cc-by-sa-3.0: Creative Commons Attribution ShareAlike 3.0 License. http://creativecommons.org/licenses/by-sa/3.0/
- cc-by-sa-2.5: Creative Commons Attribution ShareAlike 2.5 License. http://creativecommons.org/licenses/by-sa/2.5/
- cc-by-sa-2.0: Creative Commons Attribution ShareAlike 2.0 License. http://creativecommons.org/licenses/by-sa/2.0/
- cc-by-sa-1.0: Creative Commons Attribution ShareAlike 1.0 License. http://creativecommons.org/licenses/by-sa/1.0/
- cc-by-2.0: Creative Commons Attribution 2.0 License. http://creativecommons.org/licenses/by/2.0/
- cc-by-2.0: Creative Commons Attribution 2.0 License. http://creativecommons.org/licenses/by/2.0/deed.en
- cc-by-2.5: Creative Commons Attribution 2.5 License. http://creativecommons.org/licenses/by/2.5/deed.en
- cc-by-3.0: Creative Commons Attribution 3.0 License. http://creativecommons.org/licenses/by/3.0/deed.en
- GPL: GNU General Public License. http://www.gnu.org/licenses/gpl-2.0.txt
- LGPL: GNU Lesser General Public License. http://www.gnu.org/licenses/lgpl.html
- PD: This image is in the public domain.
- ATTR: The copyright holder of this file allows anyone to use it for any purpose, provided that the copyright holder is properly attributed. Redistribution, derivative work, commercial use, and all other use is permitted.
- EURO: This is the common (reverse) face of a euro coin. The copyright on the design of the common face of the euro coins belongs to the European Commission. Authorised is reproduction in a format without relief (drawings, paintings, films) provided they are not detrimental to the image of the euro.
- LFK: Lizenz Freie Kunst. http://artlibre.org/licence/lal/de
- CFR: Copyright free use.

- EPL: Eclipse Public License. `http://www.eclipse.org/org/documents/epl-v10.php`

Copies of the GPL, the LGPL as well as a GFDL are included in chapter Licenses[44]. Please note that images in the public domain do not require attribution. You may click on the image numbers in the following table to open the webpage of the images in your webbrower.

44 Chapter 27 on page 249

45 https://en.wikibooks.org/wiki/User:JMRyan
46 https://en.wikibooks.org/wiki/
47 https://en.wikibooks.org/wiki/User:JMRyan
48 https://en.wikibooks.org/wiki/
49 https://en.wikibooks.org/wiki/User:JMRyan
50 https://en.wikibooks.org/wiki/
51 https://en.wikibooks.org/wiki/User:JMRyan
52 https://en.wikibooks.org/wiki/
53 https://en.wikibooks.org/wiki/User:JMRyan
54 https://en.wikibooks.org/wiki/
55 https://en.wikibooks.org/wiki/User:JMRyan
56 https://en.wikibooks.org/wiki/
57 https://en.wikibooks.org/wiki/User:JMRyan
58 https://en.wikibooks.org/wiki/
59 https://en.wikibooks.org/wiki/User:JMRyan
60 https://en.wikibooks.org/wiki/
61 https://en.wikibooks.org/wiki/User:JMRyan
62 https://en.wikibooks.org/wiki/
63 https://en.wikibooks.org/wiki/User:JMRyan
64 https://en.wikibooks.org/wiki/
65 https://en.wikibooks.org/wiki/User:JMRyan
66 https://en.wikibooks.org/wiki/
67 https://en.wikibooks.org/wiki/User:JMRyan
68 https://en.wikibooks.org/wiki/
69 https://en.wikibooks.org/wiki/User:JMRyan
70 https://en.wikibooks.org/wiki/
71 https://en.wikibooks.org/wiki/User:JMRyan
72 https://en.wikibooks.org/wiki/
73 https://en.wikibooks.org/wiki/User:JMRyan
74 https://en.wikibooks.org/wiki/
75 https://en.wikibooks.org/wiki/User:JMRyan
76 https://en.wikibooks.org/wiki/
77 https://en.wikibooks.org/wiki/User:JMRyan
78 https://en.wikibooks.org/wiki/

79 https://en.wikibooks.org/wiki/User:JMRyan
80 https://en.wikibooks.org/wiki/
81 https://en.wikibooks.org/wiki/User:JMRyan
82 https://en.wikibooks.org/wiki/
83 https://en.wikibooks.org/wiki/User:JMRyan
84 https://en.wikibooks.org/wiki/
85 https://en.wikibooks.org/wiki/User:JMRyan
86 https://en.wikibooks.org/wiki/
87 https://en.wikibooks.org/wiki/User:JMRyan
88 https://en.wikibooks.org/wiki/
89 https://en.wikibooks.org/wiki/User:JMRyan
90 https://en.wikibooks.org/wiki/
91 https://en.wikibooks.org/wiki/User:JMRyan
92 https://en.wikibooks.org/wiki/
93 https://en.wikibooks.org/wiki/User:JMRyan
94 https://en.wikibooks.org/wiki/
95 https://en.wikibooks.org/wiki/User:JMRyan
96 https://en.wikibooks.org/wiki/
97 https://en.wikibooks.org/wiki/User:JMRyan
98 https://en.wikibooks.org/wiki/
99 https://en.wikibooks.org/wiki/User:JMRyan
100 https://en.wikibooks.org/wiki/
101 https://en.wikibooks.org/wiki/User:JMRyan
102 https://en.wikibooks.org/wiki/
103 https://en.wikibooks.org/wiki/User:JMRyan
104 https://en.wikibooks.org/wiki/
105 https://en.wikibooks.org/wiki/User:JMRyan
106 https://en.wikibooks.org/wiki/
107 https://en.wikibooks.org/wiki/User:JMRyan
108 https://en.wikibooks.org/wiki/
109 https://en.wikibooks.org/wiki/User:JMRyan
110 https://en.wikibooks.org/wiki/
111 https://en.wikibooks.org/wiki/User:JMRyan
112 https://en.wikibooks.org/wiki/

113 https://en.wikibooks.org/wiki/User:JMRyan
114 https://en.wikibooks.org/wiki/
115 https://en.wikibooks.org/wiki/User:JMRyan
116 https://en.wikibooks.org/wiki/
117 https://en.wikibooks.org/wiki/User:JMRyan
118 https://en.wikibooks.org/wiki/
119 https://en.wikibooks.org/wiki/User:JMRyan
120 https://en.wikibooks.org/wiki/
121 https://en.wikibooks.org/wiki/User:JMRyan
122 https://en.wikibooks.org/wiki/
123 https://en.wikibooks.org/wiki/User:JMRyan
124 https://en.wikibooks.org/wiki/
125 https://en.wikibooks.org/wiki/User:JMRyan
126 https://en.wikibooks.org/wiki/
127 https://en.wikibooks.org/wiki/User:JMRyan
128 https://en.wikibooks.org/wiki/
129 https://en.wikibooks.org/wiki/User:JMRyan
130 https://en.wikibooks.org/wiki/
131 https://en.wikibooks.org/wiki/User:JMRyan
132 https://en.wikibooks.org/wiki/
133 https://en.wikibooks.org/wiki/User:JMRyan
134 https://en.wikibooks.org/wiki/
135 https://en.wikibooks.org/wiki/User:JMRyan
136 https://en.wikibooks.org/wiki/
137 https://en.wikibooks.org/wiki/User:JMRyan
138 https://en.wikibooks.org/wiki/
139 https://en.wikibooks.org/wiki/User:JMRyan
140 https://en.wikibooks.org/wiki/
141 https://en.wikibooks.org/wiki/User:JMRyan
142 https://en.wikibooks.org/wiki/
143 https://en.wikibooks.org/wiki/User:JMRyan
144 https://en.wikibooks.org/wiki/
145 https://en.wikibooks.org/wiki/User:JMRyan
146 https://en.wikibooks.org/wiki/

147 https://en.wikibooks.org/wiki/User:JMRyan
148 https://en.wikibooks.org/wiki/
149 https://en.wikibooks.org/wiki/User:JMRyan
150 https://en.wikibooks.org/wiki/
151 https://en.wikibooks.org/wiki/User:JMRyan
152 https://en.wikibooks.org/wiki/
153 https://en.wikibooks.org/wiki/User:JMRyan
154 https://en.wikibooks.org/wiki/
155 https://en.wikibooks.org/wiki/User:JMRyan
156 https://en.wikibooks.org/wiki/
157 https://en.wikibooks.org/wiki/User:JMRyan
158 https://en.wikibooks.org/wiki/

27 Licenses

27.1 GNU GENERAL PUBLIC LICENSE

Version 3, 29 June 2007

Copyright © 2007 Free Software Foundation, Inc. <http://fsf.org/>

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed. Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program–to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow. TERMS AND CONDITIONS 0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion. 1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work. 2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary. 3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures. 4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee. 5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

* a) The work must carry prominent notices stating that you modified it, and giving a relevant date. * b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices". * c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it. * d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate. 6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

* a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange. * b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge. * c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b. * d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements. * e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying. 7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

* a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or * b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or * c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or * d) Limiting the use for publicity purposes of names of licensors or authors of the material; or * e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or * f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way. 8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10. 9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so. 10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it. 11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law. 12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program. 13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such. 14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License “or any later version” applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy’s public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version. 15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION. 16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. 17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the “copyright” line and a pointer to where the full notice is found.

<one line to give the program’s name and a brief idea of what it does.> Copyright (C) <year> <name of author>

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

<program> Copyright (C) <year> <name of author> This program comes with ABSOLUTELY NO WARRANTY; for details type ‘show w’. This is free software, and you are welcome to redistribute it under certain conditions; type ‘show c’ for details.

The hypothetical commands ‘show w’ and ‘show c’ should show the appropriate parts of the General Public License. Of course, your program’s commands might be different; for a GUI interface, you would use an “about box”.

You should also get your employer (if you work as a programmer) or school, if any, to sign a “copyright disclaimer” for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see <http://www.gnu.org/licenses/>.

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read <http://www.gnu.org/philosophy/why-not-lgpl.html>.

27.2 GNU Free Documentation License

Version 1.3, 3 November 2008

Copyright © 2000, 2001, 2002, 2007, 2008 Free Software Foundation, Inc. <http://fsf.org/>

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed. 0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document ”free” in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of ”copyleft”, which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference. 1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The ”Document”, below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as ”you”. You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A ”Modified Version” of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A ”Secondary Section” is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document’s overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The ”Invariant Sections” are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The ”Cover Texts” are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A ”Transparent” copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not ”Transparent” is called ”Opaque”.

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The ”Title Page” means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, ”Title Page” means the text near the most prominent appearance of the work’s title, preceding the beginning of the body of the text.

The ”publisher” means any person or entity that distributes copies of the Document to the public.

A section ”Entitled XYZ” means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as ”Acknowledgements”, ”Dedications”, ”Endorsements”, or ”History”.) To ”Preserve the Title” of such a section when you modify the Document means that it remains a section ”Entitled XYZ” according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License. 2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies. 3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document’s license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document. 4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

* A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission. * B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement. * C. State on the Title page the name of the publisher of the Modified Version, as the publisher. * D. Preserve all the copyright notices of the Document. * E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices. * F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below. * G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document’s license notice. * H. Include an unaltered copy of this License. * I. Preserve the section Entitled ”History”, Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled ”History” in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence. * J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the ”History” section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission. * K. For any section Entitled ”Acknowledgements” or ”Dedications”, Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein. * L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles. * M. Delete any section Entitled ”Endorsements”. Such a section may not be included in the Modified Version. * N. Do not retitle any existing section to be Entitled ”Endorsements” or to conflict in title with any Invariant Section. * O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version’s license notice. These titles must be distinct from any other section titles.

You may add a section Entitled ”Endorsements”, provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version. 5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled ”History” in the various original documents, forming one section Entitled ”History”; likewise combine any sections Entitled ”Acknowledgements”, and any sections Entitled ”Dedications”. You must delete all sections Entitled ”Endorsements”. 6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document. 7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an ”aggregate” if the copyright resulting from the compilation is not used to limit the legal rights of the compilation’s users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document’s Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate. 8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled ”Acknowledgements”, ”Dedications”, or ”History”, the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title. 9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it. 10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License ”or any later version” applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the Document specifies that a proxy can decide which future versions of this License can be used, that proxy’s public statement of acceptance of a version permanently authorizes you to choose that version for the Document. 11. RELICENSING

”Massive Multiauthor Collaboration Site” (or ”MMC Site”) means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A ”Massive Multiauthor Collaboration” (or ”MMC”) contained in the site means any set of copyrightable works thus published on the MMC site.

”CC-BY-SA” means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.

”Incorporate” means to publish or republish a Document, in whole or in part, as part of another Document.

An MMC is ”eligible for relicensing” if it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.

The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing. ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

Copyright (C) YEAR YOUR NAME. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled ”GNU Free Documentation License”.

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the ”with … Texts.” line with this:

with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

27.3 GNU Lesser General Public License

GNU LESSER GENERAL PUBLIC LICENSE

Version 3, 29 June 2007

Copyright © 2007 Free Software Foundation, Inc. <http://fsf.org/>

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

This version of the GNU Lesser General Public License incorporates the terms and conditions of version 3 of the GNU General Public License, supplemented by the additional permissions listed below. 0. Additional Definitions.

As used herein, "this License" refers to version 3 of the GNU Lesser General Public License, and the "GNU GPL" refers to version 3 of the GNU General Public License.

"The Library" refers to a covered work governed by this License, other than an Application or a Combined Work as defined below.

An "Application" is any work that makes use of an interface provided by the Library, but which is not otherwise based on the Library. Defining a subclass of a class defined by the Library is deemed a mode of using an interface provided by the Library.

A "Combined Work" is a work produced by combining or linking an Application with the Library. The particular version of the Library with which the Combined Work was made is also called the "Linked Version".

The "Minimal Corresponding Source" for a Combined Work means the Corresponding Source for the Combined Work, excluding any source code for portions of the Combined Work that, considered in isolation, are based on the Application, and not on the Linked Version.

The "Corresponding Application Code" for a Combined Work means the object code and/or source code for the Application, including any data and utility programs needed for reproducing the Combined Work from the Application, but excluding the System Libraries of the Combined Work. 1. Exception to Section 3 of the GNU GPL.

You may convey a covered work under sections 3 and 4 of this License without being bound by section 3 of the GNU GPL. 2. Conveying Modified Versions.

If you modify a copy of the Library, and, in your modifications, a facility refers to a function or data to be supplied by an Application that uses the facility (other than as an argument passed when the facility is invoked), then you may convey a copy of the modified version:

* a) under this License, provided that you make a good faith effort to ensure that, in the event an Application does not supply the function or data, the facility still operates, and performs whatever part of its purpose remains meaningful, or * b) under the GNU GPL, with none of the additional permissions of this License applicable to that copy.

3. Object Code Incorporating Material from Library Header Files.

The object code form of an Application may incorporate material from a header file that is part of the Library. You may convey such object code under terms of your choice, provided that, if the incorporated material is not limited to numerical parameters, data structure layouts and accessors, or small macros, inline functions and templates (ten or fewer lines in length), you do both of the following:

* a) Give prominent notice with each copy of the object code that the Library is used in it and that the Library and its use are covered by this License. * b) Accompany the object code with a copy of the GNU GPL and this license document.

4. Combined Works.

You may convey a Combined Work under terms of your choice that, taken together, effectively do not restrict modification of the portions of the Library contained in the Combined Work and reverse engineering for debugging such modifications, if you also do each of the following:

* a) Give prominent notice with each copy of the Combined Work that the Library is used in it and that the Library and its use are covered by this License. * b) Accompany the Combined Work with a copy of the GNU GPL and this license document. * c) For a Combined Work that displays copyright notices during execution, include the copyright notice for the Library among these notices, as well as a reference directing the user to the copies of the GNU GPL and this license document. * d) Do one of the following: o 0) Convey the Minimal Corresponding Source under the terms of this License, and the Corresponding Application Code in a form suitable for, and under terms that permit, the user to recombine or relink the Application with a modified version of the Linked Version to produce a modified Combined Work, in the manner specified by section 6 of the GNU GPL for conveying Corresponding Source. o 1) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (a) uses at run time a copy of the Library already present on the user's computer system, and (b) will operate properly with a modified version of the Library that is interface-compatible with the Linked Version. * e) Provide Installation Information, but only if you would otherwise be required to provide such information under section 6 of the GNU GPL, and only to the extent that such information is necessary to install and execute a modified version of the Combined Work produced by recombining or relinking the Application with a modified version of the Linked Version. (If you use option 4d0, the Installation Information must accompany the Minimal Corresponding Source and Corresponding Application Code. If you use option 4d1, you must provide the Installation Information in the manner specified by section 6 of the GNU GPL for conveying Corresponding Source.)

5. Combined Libraries.

You may place library facilities that are a work based on the Library side by side in a single library together with other library facilities that are not Applications and are not covered by this License, and convey such a combined library under terms of your choice, if you do both of the following:

* a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities, conveyed under the terms of this License. * b) Give prominent notice with the combined library that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

6. Revised Versions of the GNU Lesser General Public License.

The Free Software Foundation may publish revised and/or new versions of the GNU Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library as you received it specifies that a certain numbered version of the GNU Lesser General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that published version or of any later version published by the Free Software Foundation. If the Library as you received it does not specify a version number of the GNU Lesser General Public License, you may choose any version of the GNU Lesser General Public License ever published by the Free Software Foundation.

If the Library as you received it specifies that a proxy can decide whether future versions of the GNU Lesser General Public License shall apply, that proxy's public statement of acceptance of any version is permanent authorization for you to choose that version for the Library.