Mathematical Logic

A summary of "A Concise Introduction to Mathematical Logic", W. Rautenberg

Ari Feiglin

Contents

1 Propositional Logic

1.1 Semantics of Propositional Logic

Propositional logic is the study of logic removed from interpretation of individual variables and context. I will assume that the reader already has experience with propositional logic, as this is something an undergraduate will cover in one of their first courses. While this subsection will focus mainly on the semantics of propositional logic, we will begin by defining its *syntax*,

1.1.1 Definition

Let PV be an arbitrary set of **propositional variables** (which are regarded as arbitrary symbols). **Propositional formulas** are formulas defined recursively by the following rules,

(1) Propositional variables in PV are formulas, called **prime** or **atomic** formulas.

(2) If α and β are formulas, then so are $(\alpha \wedge \beta)$, $(\alpha \vee \beta)$, and $\neg\alpha$. $(\alpha \wedge \beta)$ is called the **conjunction** of α and β, $(\alpha \vee \beta)$ their **disjunction**, and $\neg\alpha$ the **negation** of α.

The set of all the formulas constructed in this matter is denoted $\mathcal{F}$.

We can generalize this definition; instead of utilizing only the symbols $\wedge$ and $\vee$, we can take a general *logical signature* σ consisting of logical connectives of differing arities. We then recursively define σ-formulas as following: if $\mathtt{c}$ is an n-ary logical connective in σ, and $\alpha_1, \ldots, \alpha_n$ are formulas, then so is

$$(\mathtt{c}\alpha_1, \ldots, \alpha_n)$$

Aleternatively, if we only consider binary and unary connectives, then if $\mathtt{c}$ is a unary connective, we define $\mathtt{c}\alpha$ to be a formula, and if $\circ$ is a binary connective, then $(\alpha \circ \beta)$ is a formula. But we don't have much need for such generalizations, as $\{\wedge, \vee, \neg\}$ is complete, in the sense that all connectives can be defined using them. This is a fact we will discuss soon.

We can define other connectives, for example $\rightarrow$ and $\leftrightarrow$ are used as shorthands:

$$(\alpha \rightarrow \beta) := \neg(\alpha \wedge \neg\beta), \qquad (\alpha \leftrightarrow \beta) := \big((\alpha \rightarrow \beta) \wedge (\beta \rightarrow \alpha)\big)$$

We similarly define symbols for false and true:

$$\bot := (p_1 \wedge \neg p_1), \qquad \top = \neg\bot$$

For readability, we will use the following conventions when writing formulas (this is not a change to the definition of a formula, rather conventions for writing them in order to enhance readability)

(1) We will omit the outermost parentheses when writing formulas, if there are any.

(2) The order of operations for logical connectives is as follows, from first to last: $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$.

(3) We associate $\rightarrow$ from the right, meaning $\alpha \rightarrow \beta \rightarrow \gamma$ is to be read as $\alpha \rightarrow (\beta \rightarrow \gamma)$. All other connectives associate from the left, for example $\alpha \wedge \beta \wedge \gamma$ is to be read as $(\alpha \wedge \beta) \wedge \gamma$.

(4) Instead of writing $\alpha_0 \wedge \alpha_1 \wedge \cdots \alpha_n$, we write $\bigwedge_{i=0}^{n} \alpha_i$, similar for $\vee$.

Since formulas are constructed in a recursive manner, most of our proofs about them are inductive.

1.1.2 Principle (Principle of Formula Induction)

Let $\mathcal{E}$ be a property of strings which satisfies the following conditions:

(1) $\mathcal{E}\pi$ for all prime formulas π,

(2) If $\mathcal{E}\alpha$ and $\mathcal{E}\beta$, then $\mathcal{E}(\alpha \wedge \beta)$, $\mathcal{E}(\alpha \vee \beta)$, and $\mathcal{E}\neg\alpha$ for all formulas $\alpha, \beta \in \mathcal{F}$.

Then $\mathcal{E}\varphi$ is true for all formulas φ.

An example of this is that every formula $\varphi \in \mathcal{F}$ is either prime, or of one of the following forms

$$\varphi = \neg\alpha, \quad \varphi = (\alpha \wedge \beta), \quad \varphi = (\alpha \vee \beta)$$

The proof of this is straightforward: let $\mathcal{E}$ be this property. Then trivially, $\mathcal{E}\pi$ for all prime formulas π. And if $\mathcal{E}\alpha$ and $\mathcal{E}\beta$, then of course we have

$$\mathcal{E}\neg\alpha, \quad \mathcal{E}(\alpha \wedge \beta), \quad \mathcal{E}(\alpha \vee \beta)$$

This is the first step in showing the *unique formula reconstruction property*. Let us prove a lemma before proving the property itself,

1.1.3 Lemma
Proper initial segments of formulas are not formulas. Equivalently (by contrapositive), if α and β are formulas and $\alpha\xi = \beta\eta$ for arbitrary strings ξ and η, then $\alpha = \beta$.

Let us prove this by induction on α. If α is a prime formula, suppose that β is not a prime formula, then its first character is either (or $\neg$, but then $\alpha =$ (or $\alpha = \neg$, in contradiction. Thus β is a prime formula and so $\alpha = \beta$ as they are both a single character. Now if $\alpha = (\alpha_1 \circ \alpha_2)$, then the first character of β must too be (, so β is of the form $(\beta_1 * \beta_2)$. Thus

$$\alpha_1 \circ \alpha_2)\xi = \beta_1 * \beta_2)\eta$$

and so by our inductive assumption, $\alpha_1 = \beta_1$, and so $\circ = *$, and thus $\alpha_2 = \beta_2$ by our inductive assumption again. And so $\alpha = \beta$ as required. The proof for the case that $\alpha = \neg\alpha'$ is similar. ■

1.1.4 Proposition (Unique Formula Reconstruction Property)
Every compound formula $\varphi \in \mathcal{F}$ is of one of the following forms:

$$\varphi = \neg\alpha, \quad \varphi = (\alpha \wedge \beta), \quad \varphi = (\alpha \vee \beta)$$

For some formulas $\alpha, \beta \in \mathcal{F}$.

We have already shown existence. We will now show that this is unique, meaning that φ can be written uniquely as one of these strings. Using the lemma proven above, the proof for uniqueness of the reconstruction property is immediate. For example, if $\varphi = (\alpha_1 \wedge \beta_1)$ then obviously φ cannot be written as $\neg\alpha_2$ since ($\neq \neg$, and if $\varphi = (\alpha_2 \vee \beta_2)$ then by the lemma $\alpha_1 = \alpha_2$, and so we get that $\wedge = \vee$ in contradiction. And finally if $\varphi = (\alpha_2 \wedge \beta_2)$, then again by the lemma, $\alpha_1 = \alpha_2$ and $\beta_1 = \beta_2$ as required. The proof for $\neg$ and $\vee$ are similar. ■

Utilizing formula recursion, we can define functions on formulas. For example,

1.1.5 Definition
For a formula φ, we define $\mathrm{Sf}\varphi$ to be the set of all subformulas of φ. This is done recursively:

$$\mathrm{Sf}\pi = \{\pi\} \text{ for prime formulas } \pi,$$
$$\mathrm{Sf}\neg\alpha = \mathrm{Sf}\alpha \cup \{\alpha\}, \quad \mathrm{Sf}(\alpha \circ \beta) = \mathrm{Sf}\alpha \cup \mathrm{Sf}\beta \cup \{(\alpha \circ \beta)\} \text{ for a binary logical connective } \circ$$

Similarly, we can define the **rank** of a formula φ,

$$\mathrm{rank}\pi = 0 \text{ for prime formulas } \pi,$$
$$\mathrm{rank}\neg\alpha = \mathrm{rank}\alpha + 1, \quad \mathrm{rank}(\alpha \circ \beta) = \max\{\mathrm{rank}\alpha, \mathrm{rank}\beta\} + 1 \text{ for a binary logical connective } \circ$$

And we can also define the set of variables in φ,

$$\mathrm{Var}\pi = \{\pi\} \text{ for prime formulas } \pi,$$
$$\mathrm{Var}\neg\alpha = \mathrm{Var}\alpha, \quad \mathrm{Var}(\alpha \circ \beta) = \mathrm{Var}\alpha \cup \mathrm{Var}\beta \text{ for a binary logical connective } \circ$$

In all definitions $\circ$ is either $\wedge$ or $\vee$.

So now that we have discussed the syntax of propositional logic, it is time to discuss its semantics; how we assign to formulas truth values. Recall the truth tables for $\wedge$, $\vee$, and $\neg$:

α	β	$\alpha \wedge \beta$
1	1	1
1	0	0
0	1	0
0	0	0

α	β	$\alpha \vee \beta$
1	1	1
1	0	1
0	1	1
0	0	0

α	$\neg\alpha$
1	0
0	1

These define how the logical connectives function as functions on $\{0, 1\}$.

1.1.6 Definition
A **propositional valuation**, or a **propositional model**, is a function

$$w\colon PV \longrightarrow \{0,1\}$$

We can extend it to a function $w\colon PV \longrightarrow \mathcal{F}$ as follows:

$$w(\alpha \wedge \beta) = w\alpha \wedge w\beta, \quad w(\alpha \vee \beta) = w\alpha \vee w\beta, \quad w\neg\alpha = \neg w\alpha$$

Notice that we would need to define, for example, $w(\alpha \to \beta) = w\alpha \to w\beta$ had $\to$ been an element of our logical signature. But since $\to$ is defined using $\wedge$ and $\neg$, we must prove this identity:

$$w(\alpha \to \beta) = w\neg(\alpha \wedge \neg\beta) = \neg w(\alpha \wedge \neg\beta) = \neg(w\alpha \wedge \neg w\beta) = w\alpha \to w\beta$$

This is of course not a coincidence, but a result of the fact that $\alpha \to \beta = \neg(\alpha \wedge \neg\beta)$ (where $\alpha, \beta \in \{0,1\}$). Notice that furthermore,

$$w\top = 1, \quad w\bot = 0$$

1.1.7 Proposition
The valuation of a formula is dependent only on its variables. Meaning if φ is a formula and w and w' are two valuations where $w\pi = w'\pi$ for all $\pi \in \mathrm{Var}\varphi$, then $w\varphi = w'\varphi$.

We will prove this by induction on φ. For prime formulas, this is obvious as $\mathrm{Var}\varphi = \{\varphi\}$ and then $w\varphi = w'\varphi$ by the proposition's assumption. For $\varphi = \alpha \wedge \beta$, we have that

$$w\varphi = w\alpha \wedge w\beta = w'\alpha \wedge w'\beta = w'\varphi$$

where the second equality is our inductive assumption. The proof for $\varphi = \alpha \vee \beta$ and $\varphi = \neg\alpha$ is similar.

■

Let us suppose that $PV = \{p_1, p_2, \ldots, p_n, \ldots\}$, then we define $\mathcal{F}_n$ to be the set of formulas φ such that $\mathrm{Var}\varphi \subseteq \{p_1, \ldots, p_n\}$.

1.1.8 Definition
A **boolean function** is a function

$$f\colon \{0,1\}^n \longrightarrow \{0,1\}$$

for some $n \geq 0$. The set of boolean functions of arity n is denoted $\mathbf{B}_n$. A formula $\varphi \in \mathcal{F}_n$ **represents** a boolean function $f \in \mathbf{B}_n$ (similarly, f is represented by φ), if for all valuations w,

$$w\varphi = f(w\vec{p}) \qquad (w\vec{p} = (wp_1, \ldots, wp_n))$$

So for example, $\alpha = p_1 \wedge p_2$ represents the function $f(p,q) = p \wedge q$. This is since

$$f(wp_1, wp_2) = wp_1 \wedge wp_2 = w(p_1 \wedge p_2) = w\alpha$$

Since valuations of $\varphi \in \mathcal{F}_n$ are defined by their values on $p_1, \ldots, p_n$, φ represents at most a single function f. In fact, it represents the function

$$\varphi^{(n)}(x_1, \ldots, x_n) = w\varphi$$

where w is any valuation such that $wp_i = x_i$ (all of these valuations valuate φ the same). Now, notice that $\mathcal{F}_n \subset \mathcal{F}_{n+1}$ and $\mathbf{B}_n \subset \mathbf{B}_{n+1}$ and so $\varphi \in \mathcal{F}_n$ represents a function in $\mathcal{B}_{n+1}$ as well. But this function is not essentially in $\mathcal{B}_n$ in the sense that its last argument does not impact its value. Formally we say that given a function $f\colon M^n \longrightarrow M$, we call its ith argument *fictional* if for all $x_1, \ldots, x_i, \ldots, x_n \in M$ and $x_i' \in M$:

$$f(x_1, \ldots, x_i, \ldots, x_n) = f(x_1, \ldots, x_i', \ldots, x_n)$$

An *essentially n-ary* function is a function with no fictional arguments.

1.1.9 Definition
Two formulas α and β are **equivalent** if for every valuation w, $w\alpha = w\beta$. This is denoted $\alpha \equiv \beta$.

It is immediate that α and β are equivalent if and only if they represent the same function. A simple example of equivalence is $\alpha \equiv \neg\neg\alpha$. The following equivalences are easy to verify and the reader should already be familiar with them (α, β, and γ are formulas):

$$\begin{array}{cc} \alpha \wedge (\beta \wedge \gamma) \equiv \alpha \wedge \beta \wedge \gamma & \alpha \vee (\beta \vee \gamma) \equiv \alpha \vee \beta\gamma \quad \text{(associativity)} \\ \alpha \wedge \beta \equiv \beta \wedge \alpha & \alpha \vee \beta \equiv \beta \vee \alpha \quad \text{(commutativity)} \\ \alpha \wedge \alpha \equiv \alpha & \alpha \vee \alpha \equiv \alpha \quad \text{(idempotency)} \\ \alpha \wedge (\alpha \vee \beta) \equiv \alpha & \alpha \vee \alpha \wedge \beta \equiv \alpha \quad \text{(absorption)} \end{array}$$

$$\alpha \wedge (\beta \vee \gamma) \equiv \alpha \wedge \beta \vee \alpha \wedge \gamma \quad (\wedge\text{-distributivity})$$

$$\alpha \vee \beta \wedge \gamma \equiv (\alpha \vee \beta) \wedge (\alpha \vee \gamma) \quad (\vee\text{-distributivity})$$

$$\neg(\alpha \wedge \beta) \equiv \neg\alpha \vee \neg\beta \qquad \neg(\alpha \vee \beta) \equiv \neg\alpha \wedge \neg\beta \quad \text{(de Morgan rules)}$$

Furthermore,

$$\alpha \vee \neg\alpha \equiv \top, \quad \alpha \wedge \neg\alpha \equiv \bot, \quad \alpha \wedge \top \equiv \alpha \vee \bot \equiv \alpha$$

Since $\alpha \to \beta = \neg(\alpha \wedge \neg\beta)$, by de Morgan rules, this is equivalent to

$$\equiv \neg\alpha \vee \neg\neg\beta \equiv \neg\alpha \vee \beta$$

Notice that

$$\alpha \to \beta \to \gamma \equiv \neg\alpha \vee (\beta \to \gamma) \equiv \neg\alpha \vee \neg\beta \vee \gamma \equiv \neg(\alpha \wedge \beta) \vee \gamma \equiv \alpha \wedge \beta \to \gamma$$

Inductively, we see that

$$\alpha_1 \to \cdots \to \alpha_n \to \gamma \equiv \alpha_1 \wedge \cdots \wedge \alpha_n \to \gamma$$

We could go on, but I assume you get the point.
$\equiv$ is obviously reflexive, symmetric, and transitive: therefore it is an equivalence relation on $\mathcal{F}$. But moreso it is a *congruence relation*, meaning it respects connectives. Explicitly, for all formulas $\alpha, \beta, \alpha', \beta' \in \mathcal{F}$:

$$\alpha \equiv \alpha',\ \beta \equiv \beta' \implies \alpha \wedge \beta \equiv \alpha' \wedge \beta',\ \alpha \vee \beta \equiv \alpha' \vee \beta',\ \neg\alpha \equiv \neg\alpha'$$

Congruence relations will be discussed in more generality in later sections. Inductively, we can prove the following result:

1.1.10 Theorem (The Replacement Theorem)
Suppose α and α' are equivalent formulas. Let φ be some other formula, and define φ' to be the result of replacing all occurences of α within φ by α'. Then $\varphi \equiv \varphi'$.

This will be proven more generally later.

1.1.11 Definition
Prime formulas and their negations are called **literals**. A formula of the form $\alpha_1 \vee \cdots \vee \alpha_n$ where each α_i is a conjunction of literals is called a **disjunctive normal form**. And similarly a formula of the form $\alpha_1 \wedge \cdots \wedge \alpha_n$ where each α_i is a disjunction of literals is called a **conjunctive normal form**. We will use the abreviations DNF and CNF for disjunctive and conjunctive normal forms, respectively.

So a DNF is a formula of the form

$$\bigvee_{i=1}^{n} \bigwedge_{j=1}^{m_i} \ell_{i,j}$$

where for every i, j, $\ell_{i,j}$ is a literal: a formula of the form $p_{i,j}$ or $\neg p_{i,j}$ for some prime formula $p_{i,j}$. Similarly a CNF is a formula of the form

$$\bigwedge_{i=1}^{n} \bigvee_{j=1}^{m_i} \ell_{i,j}$$

Let us temporarily introduce the following notation: for a prime formula p, let

$$p^1 := p, \qquad p^0 := \neg p$$

This allows us to more concisely state and prove the following theorem:

1.1.12 Theorem
Every boolean function $f \in \mathbf{B}_n$ for $n > 0$ is representable by the DNF

$$\alpha_f := \bigvee_{f(\vec{x})=1} p_1^{x_1} \wedge \cdots \wedge p_n^{x_n}$$

and a CNF

$$\beta_f := \bigwedge_{f(\vec{x})=0} p_1^{\neg x_1} \wedge \cdots \wedge p_n^{\neg x_n}$$

Let w be a valuation and $\vec{p} = (p_1, \ldots, p_n)$ then

$$w\alpha_f = \bigvee_{f(\vec{x})=1} wp_1^{x_1} \wedge \cdots \wedge wp_n^{x_n}$$

Notice that wp^x is equal to 1 if and only if $wp = x$: suppose $x = 0$ then $wp^x = \neg wp$, which is equal to 1 if and only if $wp = 0 = x$, and similar for $x = 1$. Thus $w\alpha_f = 1$ if and only if there exists a $\vec{x}$ such that $f(\vec{x}) = 1$ and $wp_1^{x_1} \wedge \cdots \wedge wp_n^{x_n} = 1$, meaning that for each i, $wp_i = x_i$. This means that $\vec{x} = w\vec{p}$, and so $f(w\vec{p}) = f(\vec{x}) = 1$. Similarly if $f(w\vec{p}) = 1$ then let $\vec{x} = w\vec{p}$, and then $wp_1^{x_1} \wedge \cdots \wedge wp_n^{x_n} = 1$ and $f(\vec{x}) = 1$, so $w\alpha_f = 1$. So $w\alpha_f = f(w\vec{p})$ for all valuations w, which means that f is represented by α_f, as required. The proof for β_f is similar. ■

Notice that since every formula represents a boolean function, which by above can be represented by a DNF and a CNF, we get that every formula is equivalent to a DNF and a CNF.

1.1.13 Corollary
Every formula is equivalent to a DNF and a CNF.

1.1.14 Definition
A logical signature σ is **functional complete** if every boolean function is representable by a formula in this signature.

By corollary 1.1.13, $\{\neg, \wedge, \vee\}$ is functional complete. Since

$$\alpha \vee \beta \equiv \neg(\neg\alpha \wedge \neg\beta), \quad \alpha \wedge \beta \equiv \neg(\neg\alpha \vee \neg\beta)$$

$\{\neg, \wedge\}$ and $\{\neg, \vee\}$ are both functional complete. Thus in order to show that a logical signature σ is functional complete, it is sufficient to show that $\neg$ and $\wedge$ or $\neg$ and $\vee$ can be represented by σ.

Note
If f is a function, instead of writing $f(x)$ or fx, many times we will instead write x^f. This is more concise and may reduce confusion in the case that x itself is a string wrapped in parentheses.

Let us define the function $\delta\colon \mathcal{F} \longrightarrow \mathcal{F}$ on formulas recursively by $p^\delta = p$ for prime formulas p and

$$(\neg\alpha)^\delta = \neg\alpha^\delta, \quad (\alpha \wedge \beta)^\delta = \alpha^\delta \vee \beta^\delta, \quad (\alpha \vee \beta)^\delta = \alpha^\delta \wedge \beta^\delta$$

Alternatively, α^δ is simply the result of swapping all occurences of $\wedge$ with $\vee$, and all occurences of $\vee$ with $\wedge$. α^δ is called the *dual formula* of α. Notice that the dual formula of a DNF is a CNF, and vice versa.
Now, suppose $f \in \mathbf{B}_n$, then let us define the *dual* of f,

$$f^\delta(\vec{x}) := \neg f(\neg\vec{x})$$

where $\neg\vec{x} = (\neg x_1, \ldots, \neg x_n)$. Notice that δ is idempotent:

$$f^{\delta^2}(\vec{x}) = \neg f^\delta(\neg\vec{x}) = \neg\neg f(\neg\neg\vec{x}) = f(\vec{x})$$

1.1.15 Theorem (The Duality Principle for Two-Valued Logic)
If α represents the function f, then α^δ represents f^δ.

We will prove this by induction on α. If $\alpha = p$ is prime, then this is trivial. Now suppose that α and β represent f_1 and f_2 respectively. Then $\alpha \wedge \beta$ represents $f(\vec{x}) = f_1(\vec{x}) \wedge f_2(\vec{x})$, and $(\alpha \wedge \beta)^\delta = \alpha^\delta \vee \beta^\delta$ represents $g(\vec{x}) = f_1^\delta(\vec{x}) \vee f_2^\delta(\vec{x})$ by the induction hypothesis. Now,

$$f^\delta(\vec{x}) = \neg f(\neg\vec{x}) = \neg(f_1(\neg\vec{x}) \wedge f_2(\neg\vec{x})) = \neg f_1(\neg\vec{x}) \vee \neg f_2(\neg\vec{x}) = f_1^\delta(\vec{x}) \vee f_2^\delta(\vec{x}) = g(\vec{x})$$

So $f^\delta = g$, meaning that $(\alpha \wedge \beta)^\delta$ does indeed represent f^δ. The proof for $\alpha \vee \beta$ is similar. Now suppose α represents f, then $\neg\alpha$ represents $\neg f$, and α^δ represents f^δ by the induction hypothesis. And so $(\neg\alpha)^\delta = \neg\alpha^\delta$ represents $\neg f^\delta$, which is equal to $(\neg f)^\delta$ since

$$(\neg f)^\delta(\vec{x}) = (\neg\neg f)(\neg\vec{x}) = \neg(\neg f(\neg\vec{x})) = \neg f^\delta(\vec{x})$$

And so $(\neg\alpha)^\delta$ represents $(\neg f)^\delta$, as required. ■

1.1.16 Definition
Suppose α is a formula and w is a valuation. Instead of writing $w\alpha = 1$, we now write $w \vDash \alpha$, and this is read as "w *satisfies* α". If X is a set of formulas, we write $w \vDash X$ if $w \vDash \alpha$ for all $\alpha \in X$, and w is called a **propositional model** for X. A formula α (respectively a set of formulas X) is **satisfiable** if there is some valuation w such that $w \vDash \alpha$ (respectively $w \vDash X$). $\vDash$ is called the **satisfiability relation**.

$\vDash$ has the following immediate properties:

$$w \vDash p \iff wp = 1 \ (p \in PV) \qquad w \vDash \alpha \iff w \nvDash \alpha$$
$$w \vDash \alpha \wedge \beta \iff w \vDash \alpha \text{ and } w \vDash \beta \qquad w \vDash \alpha \vee \beta \iff w \vDash \alpha \text{ or } w \vDash \beta$$

These properties uniquely define $\vDash$, meaning we could have defined $\vDash$ recursively by these properties.
Notice that

$$w \vDash \alpha \to \beta \iff \text{if } w \vDash \alpha \text{ then } w \vDash \beta$$

This is due to the definition of $\to$ coinciding with our common usage of implication. Had we not defined $\to$, but instead added it to our logical signature, this above equivalence would have to be taken in the definition of the satisfiability relation (when axiomized by the above properties).

1.1.17 Definition
α is **logically valid**, or a **tautology**, if $w \vDash \alpha$ for all valuations w. This is abbreviated by $\vDash \alpha$. A formula which cannot be satisfied; ie. for all valuations w, $w \nvDash \alpha$; is called a **contradiction**.

For example, $\alpha \vee \neg\alpha$ is a tautology, while $\alpha \wedge \neg\alpha$ and $\alpha \leftrightarrow \neg\alpha$ are contradictions for all formulas α. Notice that the negation of a tautology is a contradiction and vice versa. $\top$ is a tautology and $\bot$ is a contradiction. The following are important tautologies of implication (keep in mind how $\to$ associates from the right):

$p \to p$	(self-implication)
$(p \to q) \to (q \to r) \to (p \to r)$	(chain rule)
$(p \to q \to r) \to (q \to p \to r)$	(exchange of premises)
$p \to q \to p$	(premise change)
$(p \to q \to r) \to (p \to q) \to (p \to r)$	(Frege's formula)
$((p \to q) \to p) \to p$	(Peirce's formula)

1.1.18 Definition
Suppose X is a set of formulas and α a formula, we say that α is a **logical consequence** if $w \vDash \alpha$ for every model w of X. In other words,

$$w \vDash X \implies w \vDash \alpha$$

This is denoted $X \vDash \alpha$.

Notice that $\vDash$ here is used for logical consequence (the consequence relation), and we used it before as the symbol for the satisfiability relation. Context will make it clear as to its meaning. We use the notation $\alpha_1, \ldots, \alpha_n \vDash \beta$ to mean $\{\alpha_1, \ldots, \alpha_n\} \vDash \beta$. This justifies the notation for tautologies: α is a tautology if and only if $\varnothing \vDash \alpha$ (since every valuation models $\varnothing$), which is shortened by the above notation to $\vDash \alpha$.
And we also use $X \vDash \alpha, \beta$ to mean $X \vDash \alpha$ and $X \vDash \beta$. And $X, \alpha \vDash \beta$ to mean $X \cup \{\alpha\} \vDash \beta$.
The following are examples of logical consequences

$$\alpha, \beta \vDash \alpha \wedge \beta, \quad \alpha \wedge \beta \vDash \alpha, \beta$$
$$\alpha, \alpha \to \beta \vDash \beta$$
$$X \vDash \bot \implies X \vDash \alpha \quad \text{for all formulas } \alpha$$
$$X, \alpha \vDash \beta,\ X, \neg\alpha \vDash \beta \implies X \vDash \beta$$

The final example is true because if $w \vDash X$ then either $w \vDash \alpha$ or $w \vDash \neg\alpha$, and in either case $w \vDash \beta$.
Let us now state some obvious properties of the consequence relation:

$$\begin{array}{ll} \alpha \in X \implies X \vDash \alpha & (\textit{reflexivity}) \\ X \vDash \alpha \text{ and } X \subseteq X' \implies X' \vDash \alpha & (\textit{monotonicity}) \\ X \vDash Y \text{ and } Y \vDash \alpha \implies X \vDash \alpha & (\textit{transitivity}) \end{array}$$

1.1.19 Definition
A **propositional substitution** is a mapping from prime formulas to formulas, $\sigma\colon PV \longrightarrow \mathcal{F}$, which is extended to a mapping between formulas $\sigma\colon \mathcal{F} \longrightarrow \mathcal{F}$ recursively:

$$(\alpha \wedge \beta)^\sigma = \alpha^\sigma \wedge \beta^\sigma, \quad (\alpha \vee \beta)^\sigma = \alpha^\sigma \vee \beta^\sigma, \quad (\neg\alpha)^\sigma = \neg\alpha^\sigma$$

2 Index