MATH70132: Mathematical Logic

Lecturer: David Evans

Scribe: Sidharth Hariharan

Imperial College London - Spring 2025

Contents

Chapter 1

Propositional Logic

Propositional logic is the logic of reasoning and proof. Before we get started with anything formal, here's a motivating example.

Consider the following statement:

> If Mr Jones is happy, then Mrs Jones is unhappy, and if Mrs Jones is unhappy, then Mr Jones is unhappy. Therefore, Mr Jones is unhappy.

One can ask ourselves whether it is logically valid to conclude that Mr Jones is unhappy based on the relationship between the happiness of Mr Jones and that of Mrs Jones expressed in the sentence preceding it.

Putting this into symbols, let P denote the statement that Mr Jones is happy, and let Q denote the statement that Mrs Jones is unhappy. We can express the statement as follows:

$$((P \implies Q) \wedge (Q \implies \neg P)) \implies (\neg P) \tag{1.0.1}$$

This disambiguation, by removing any question of marital harmony from what is otherwise a purely logical question, allows us to manually check whether (??) is a valid statement by constructing a **truth table**.

We will begin by developing some machinery to reason about these sorts of statements more formally.

1.1 Propositional Formulae

1.1.1 Propositions and Connectives

We begin by defining the notion of a proposition.

Definition 1.1.1 (Proposition). A **proposition** is a statement that is either true or false.

Convention. We will denote the state of being **true** by **T** and that of being **false** by **F**.

Propositions can be connected to each other using tools known as **connectives**. These can be thought of as **truth table rules**.

Convention. Before we define the actual connectives we shall use, we list them down, along with notation.

1. Conjunction ($\wedge$)
2. Disjunction ($\vee$)
3. Negation ($\neg$)
4. Implication ($\rightarrow$)
5. The Biconditional ($\leftrightarrow$)

In particular, we will only use the $\implies$ and $\iff$ symbols when reasoning **informally**. For **formal** use, we will stick to the $\rightarrow$ and $\leftrightarrow$ symbols.

We define them exhaustively as follows.

Definition 1.1.2 (Connectives). Let p and q be true/false variables. We define each of the connectives listed above to take on truth values depending on those of p and q as follows.

p	q	$(\neg p)$	$(\neg q)$	$(p \wedge q)$	$(p \vee q)$	$(p \rightarrow q)$	$(p \leftrightarrow q)$
T	**T**	**F**	**F**	**T**	**T**	**T**	**T**
T	**F**	**F**	**T**	**F**	**T**	**F**	**F**
F	**T**	**T**	**F**	**F**	**T**	**T**	**F**
F	**F**	**T**	**T**	**F**	**F**	**T**	**T**

We are now ready to define the main object of study in this section: propositional formulae.

Definition 1.1.3 (Propositional Formula). A **propositional formula** is obtained from propositional variables and connectives via the following rules:

(i) Any propositional variable is a propositional formula.

(ii) If ϕ and ψ are formulae, then so are $(\neg\phi)$, $(\neg\psi)$, $(\phi \wedge \psi)$, $(\phi \vee \psi)$, $(\phi \to \psi)$, $(\psi \to \phi)$, and $(\phi \leftrightarrow \psi)$.

(iii) Any formula arises in this manner after a finite number of steps.

What this means is that a propositional formula is a string of symbols consisting of

1. variables that take on true/false values,
2. connectors that express the relationship between these variables, and
3. parentheses/brackets that separate formulae within formulae and specify the order in which they must be evaluated when the constituent variables are assigned specific values.

In particular, every propositional formula is either a propositional variable or is built from 'shorter' formulae, where by 'shorter' we mean consisting of fewer symbols.

Convention. Throughout this module, we will adopt two important conventions when dealing with propositional formulae.

1. All propositional formulae, barring those consisting of a single variable, shall be enclosed in parentheses.
2. When we want to denote a propositional formula by a certain symbol, we will use the notation "symobol : formula".

As a concluding remark on the nature of propositional formulae, we will note that just as we use trees to evaluate expressions on the computer when performing arithmetic, we can use them to express and evaluate propositional formulae as well. We will not usually do this, however, as it takes up a lot of space. In any event, we would first need to make precise the notion of *evaluating* a propositional formula. For this, we will turn to the concept of a truth function.

1.1.2 Truth Functions

Any assignment of truth values to the propositional variables in a formula ϕ determines the truth value for ϕ in a **unique** manner, using the exhaustive definitions of the connectives given in ??. We often express all possible values of a propositional formula in a **truth table**, much like we did in ?? when defining the connectives.

Example 1.1.4. Consider the formula $\phi : ((p \to (\neg q)) \to p)$, where p and q are propositional variables. We construct a truth table as follows.

p	q	$(\neg q)$	$(p \to (\neg q))$	$((p \to (\neg q)) \to p)$
T	**T**	**F**	**F**	**T**
T	**F**	**T**	**T**	**T**
F	**T**	**F**	**T**	**F**
F	**F**	**F**	**T**	**F**

From this table, it is clear that the truth value of ϕ depends on the truth values of p and q in some manner (to be perfectly precise, it only depends on the truth value of p, and is, in fact, biconditionally equivalent to p). We would like to have a formal notion of navigating this dependence to 'compute a value for ϕ given values of p and q'.

Throughout this subsection, n will denote an arbitrary natural number.

Definition 1.1.5 (Truth Function). A **truth function** of n variables is a function

$$f : \{\mathbf{T}, \mathbf{F}\}^n \to \{\mathbf{T}, \mathbf{F}\}$$

These are very directly related to propositional formulae.

Definition 1.1.6 (Truth Function of a Propositional Formula). Let ϕ be a propositional formula whose variables are $p_1, \ldots, p_n$. We can associate to ϕ a truth function whose truth value at any $(x_1, \ldots, x_n) \in \{\mathbf{T}, \mathbf{F}\}^n$ corresponds to the truth value of ϕ that arises from setting p_i to x_i for all $1 \leq i \leq n$. We define this truth function to be the **truth function of** ϕ, denoted F_ϕ.

We can now construct a truth function for the example we saw at the very beginning involving Mr and Mrs Jones (cf. (**??**)).

Example 1.1.7. `sorry`

We see something quite remarkable here: the truth function of the propositional formula defined in (**??**) maps every possible input to **T**! We have a special term for this.

Definition 1.1.8 (Tautology)**.** A propositional formula ϕ is a **tautology** if its truth function F_ϕ maps every possible input to **T**.

We can also be more precise about what the biconditional connective actually tells us.

Definition 1.1.9 (Logical Equivalence)**.** The propositional formulae ψ and χ are **logically equivalent** if the truth function $F_{\psi\leftrightarrow\chi}$ of their biconditional is a tautology.

We have a fairly basic result about logical equivalence.

Lemma 1.1.10. *Let $p_1, \ldots, p_n$ be propositional variables and let ψ and χ be formulae in $p_1, \ldots, p_n$. Then, ψ and χ are logically equivalent if and only if $F_\psi = F_\chi$.*

We omit the proof of this result as it merely involves checking things manually. A computer should be able to do this almost instantaneously.

We can also say something about composing formulae together.

Lemma 1.1.11. *Suppose that ϕ is a propositional formula with variables $p_1, \ldots, p_n$. Let $\phi_1, \ldots, \phi_n$ be propositional formulae. Denote by ϑ the result of substituting each p_i with ϕ_i in ϕ. Then,*

(i) *ϑ is a propositional formula.*

(ii) *if ϕ is a tautology, so is ϑ.*

(iii) *the truth function of ϑ is the result of composing the truth function of ϕ with the Cartesian product of the truth functions of $\phi_1, \ldots, \phi_n$.*

We do not prove this result either, as it merely involves manual verification.

Example 1.1.12. For propositional variables p_1, p_2, the statement $(((\neg p_2) \to (\neg p_1)) \to (p_1 \to p_2))$ is a tautology. Therefore, if ϕ_1 and ϕ_2 are propositional formulae, then $(((\neg \phi_2) \to (\neg \phi_1)) \to (\phi_1 \to \phi_2))$ is a tautology as well.

We will also mention that a composition being a tautology does not mean the outermost proposition of the composition is a tautology.

Non-Example 1.1.13. Let p be a propositional variable. The formula $\phi : (p \to (\neg p))$ is not a tautology. However, we can find a propositional formula ϕ' such that $(\phi_1 \to (\neg \phi_1))$ is a tautology: for example, we can define ϕ' to be identically $\mathbf{F}$.

There are numerous propositional formulae that we know to be logically equivalent. Here is a (non-exhaustive) list.

Example 1.1.14 (Logically Equivalent Formulae)**.** Let p_1, p_2, p_3 be logically equivalent formulae. Then, the following equivalences hold.

1. $(p_1 \wedge (p_2 \wedge p_3))$ is logically equivalent to $((p_1 \wedge p_2) \wedge p_3)$.
2. $(p_1 \vee (p_2 \vee p_3))$ is logically equivalent to $((p_1 \vee p_2) \vee p_3)$.
3. $(p_1 \vee (p_2 \wedge p_3))$ is logically equivalent to $((p_1 \vee p_2) \wedge (p_1 \wedge p_3))$.
4. $(\neg (\neg p_1))$ is logically equivalent to p_1.
5. $(\neg (p_1 \wedge p_2))$ is logically equivalent to $((\neg p_1) \vee (\neg p_2))$.
6. $(\neg (p_1 \vee p_2))$ is logically equivalent to $((\neg p_1) \wedge (\neg p_2))$.

Upon inspection, one can find algebraic patterns in the above logical equivalences. There are similarities to the axioms of a **boolean algebra**. We will not explore this further in this module, but we will adopt the convention used in algebra where parentheses are dropped when dealing with associative operations.

Convention. We will denote both $(p_1 \wedge (p_2 \wedge p_3))$ and $((p_1 \wedge p_2) \wedge p_3)$ by $(p_1 \wedge p_2 \wedge p_3)$. Similarly, we will denote both $(p_1 \vee (p_2 \vee p_3))$ and $((p_1 \vee p_2) \vee p_3)$ by $(p_1 \vee p_2 \vee p_3)$.

We will end with a combinatorial fact about truth functions.

Lemma 1.1.15. *There are 2^{2^n} possible truth functions on n variables.*

Proof. A truth function is any function from a the set $\{\mathbf{T}, \mathbf{F}\}^n$ to the set $\{\mathbf{T}, \mathbf{F}\}$, with no further restrictions. The former set has 2^n elements and the latter set has 2 elements. Therefore, there are 2^{2^n} possible truth functions. □

1.1.3 More about Connectives

We have defined several connectives so far, but we have yet to say anything about whether we will be defining any more connectives going forward. To begin, we will state an important definition.

Definition 1.1.16 (Adequacy). We say that a set S of connectives is **adequate** if for every $n \geq 1$, every truth function on n variables can be expressed as the truth function as a propositional formula which only involves connectives from S (and n propositional variables).

We now have the first theorem of this module.

Theorem 1.1.17. *The set $\{\neg, \wedge, \vee\}$ is adequate.*

Proof. Fix some $n \geq 1$, and let $G : \{\mathbf{T}, \mathbf{F}\}^n \to \{\mathbf{T}, \mathbf{F}\}$ be a truth function. We have two cases.

Case 1. Assume that G is identically $\mathbf{F}$. Then, define the propositional formula $\phi : (p_1 \wedge (\neg p_1))$. Even defining it as a formula on n variables, it is clear to see that its truth function F_ϕ is identically $\mathbf{F}$. Therefore, $G = F_\phi$.[1]

Case 2. Assume the contrary. Then, list all $v \in \{\mathbf{T}, \mathbf{F}\}^n$ such that $G(v) = \mathbf{T}$. Since $\{\mathbf{T}, \mathbf{F}\}^n$ is a finite set, this list is finite, and we can number these $v_1, \ldots, v_r$. For each $1 \leq i \leq r$, denote

$$v_i = (v_{i1}, \ldots, v_{ir})$$

where $v_{ij} \in \{\mathbf{T}, \mathbf{F}\}$ is the jth component of v_i. Let $p_1, \ldots, p_n$ be propositional variables. Define the propositional formulae `sorry`

□

[1] Admittedly, we are using the Axiom of Extensionality here to define what it means for the two functions to be equal. We will ignore this technicality for now.

Here, we make an important definition.

Definition 1.1.18 (Disjunctive Normal Form). When a propositional formula is expressed only in terms of propositional variables and the set $\{\neg, \wedge, \vee\}$ of connectives, it is said to be in **disjunctive normal form**, which we abbreviate to **DNF**.

The point of **??** is the following.

Corollary 1.1.19. *Every propositional formula is expressible in DNF.*

Proof. We know that every propositional formula admits a truth function. For any propositional formula, we can apply **??** to its truth function. Then, unfolding the definition of adequacy yields the desired result. □

Example 1.1.20. Let p_1 and p_2 be propositional variables. Consider the propositional formula $\chi : ((p_1 \to p_2) \to (\neg p_2))$. We can see that $F_\chi(v) = \mathbf{T}$ only if $v = (\mathbf{T}, \mathbf{F})$ or $v = (\mathbf{F}, \mathbf{F})$. Therefore, the DNF of χ is

$$((p_1 \wedge (\neg p_2)) \vee ((\neg p_1) \wedge (\neg p_2)))$$

Despite the result about DNF, it turns out there are other sets of connectives that are adequate.

Lemma 1.1.21. *The following sets of connectives are adequate.*

1. $\{\neg, \vee\}$
2. $\{\neg, \wedge\}$
3. $\{\neg, \to\}$

For the latest version of these notes, visit https://github.com/thefundamentaltheor3m/LieAlgebrasNotes. For any suggestions or corrections, please feel free to fork and make a pull request to my repository.