CHAPTER 2

Learning how to count and reason (Cont'd)

Your first night in Hilbert's Hotel started in a rather eventful manner, but now that the passengers of Peano's Plane are all settled in their rooms and the passengers of *The Paradise* are looking around for somewhere else to sleep, you're free to relax a bit. You spent some time counting sheep – enough time to get all the way up to $2^{\aleph_0}$, but alas! Still, you cannot sleep.

You think to yourself:

- If I could sleep, then tomorrow I'd be well-rested.
- If tomorrow I'm well-rested, then I will have a productive day.
- If tomorrow I have a productive day, then I'll be happy.

Unfortunately, as we've already established, you cannot sleep. The thought that, therefore, tomorrow you will be unhappy keeps plaguing your mind. Thankfully(?) Hilbert's Hotel just so happens to be haunted, and a strange ghost covered in rain happens to be haunting your room. It is, of course, the ghost of George Boole, who is extremely annoyed about your train of thought. So annoyed, in fact, that he decides to take a break from purgatory and make himself useful.

2. Things are either true or false

We'll spend the rest of this chapter learning how to reason. We'll first learn how to reason in *propositional logic*, just to warm up for the more grown-up *first-order logic*, which is our main topic (and is usually what logicians mean when they talk about "logic"). The examples from Chapter 1 involved some kind of quantification ("all cops" – that is the first-order part),[1] but there are many interesting examples to discuss that don't.

[1]The classical example that philosophers love to quote is the following: "*'All men are mortal'*; *'Socrates is a man'*; THEREFORE *'Socrates is mortal'*". I'm not sure why this example is so overused, but I'd feel weird not including it somewhere in this text. Maybe I'm part of the problem? If you're interested, this formulation is actually due to John Stuart Mill (1843).

One is the sequence of implications from a couple of paragraphs above. Here are two more:

GIVEN:

- P_1: When a stranger enters the house, my cats meow.
- P_2: There is a thief in the living room.

DEDUCE: D: My cats are mewing.

That's some solid reasoning! What about the following?

GIVEN:

- S_1 : If I have my umbrella with me, then I don't get wet.
- S_2: I am dry.

DEDUCE: D': I had my umbrella.

Again, the second deduction really does not feel all that good, does it?

Our goal for now will be to abstract all distractions away and try to understand exactly what the distinction between a valid and an invalid propositional argument is. The first distraction is all the words describing the situations in the arguments ("a stranger enters the house", "I have my umbrella", etc.) and the second distraction is all those connecting words ("when", "if", "then", etc.). Doing as mathematicians do, we will replace them with symbols that have precise meanings.

There is an important distinction between *syntax* (how things are written down) and *semantics* (what things mean). Let's not get *too* bugged down with this yet, but as a warning, the closer we get to incompleteness, the more important this distinction will get. Let's start with the symbols that we will be using to abstract away our words. We have the following **logical connectives**:

- $\wedge$ (syntactically *"wedge"*, semantically *"and"*);
- $\vee$ (syntactically *"vee"*, semantically *"and"*);
- $\rightarrow$ (syntactically *"right arrow"*, semantically *"implies"*);
- $\neg$ (syntactically *"I'm sure this has a name, but nobody has told me what it is yet"*, semantically *"not"*);

and if we're being very pedantic, we should also include parentheses (i.e. the symbols ")" and "(") in our list of symbols too.

Exercise 2.0.1. Given the intuitive meaning of the logical connectives, try to use them (in whatever way you see fit) to express the arguments you've seen so far in a mathematical way.

If you gave the previous exercise a try you will probably have realised that we still need actual precise rules if we want to be able to play this "logic" game. That's what we'll get to next.

2.1. Formulas. To abstract the particulars of given situations (e.g. the meowing of cats or the presence of an umbrella) away and focus on the structure of the arguments, we will keep a collection of variables, which we refer to as **propositional variables**. We will write Var for the set of all propositional variables. Now that you know all about cardinality, we will assume that $|\mathsf{Var}| = \aleph_0$.[2]

You can think of the elements of Var as stand-ins for English sentences, in the same way that variables in algebra are stand-ins for numbers. In this chapter, I will mainly be using the letters $A, A_1, A_2, \ldots$, and $B, B_1, B_2, \ldots$, etc. to denote propositional variables.

A **formula of propositional logic** (some people that are certainly more pedantic than I am[3] call these *well-formed formulas*, but like if a formula is not well-formed, then it's not really a formula, is it?) is built, inductively, as follows:

(1) A propositional variable is a formula.

(2) If ϕ and ψ are formulas, then so are:

   (a) $(\phi \wedge \psi)$

   (b) $(\phi \vee \psi)$

   (c) $(\phi \to \psi)$

   (d) $(\neg\phi)$

(3) Every propositional formula is finite and built this way.

Remark 2.1.1. Strictly speaking, (3) in the definition above is not necessary, but it sort of maybe clarifies the picture a tad. It says that the set of all propositional formulas is the smallest set of finite strings from the list symbols in the previous

[2]This is mainly to make our lives easier, in some proofs. You may have read a bit about the axiom of choice in the blue(=optional) part of last week's material, and even though it makes me blue(=sad), I'd like to keep it blue(=optional).

[3]Yes, such people do exist.

page that contains all the propositional variables, and is closed under the operations in (2).

For now, propositional logic is all we have, so I will be referring to formulas of propositional logic as simply *formulas*.

Example 2.1.2. Here are some examples of formulas using these symbols: $(A \wedge B)$, $(A \vee B)$, $((\neg(A \wedge B)) \rightarrow ((\neg A) \vee (\neg B)))$. On the other hand, $\wedge AB$ is not a formula (unless you're Polish), and neither is $\wedge A \vee B \rightarrow$ (not even in Poland).

For now (though not for long), these formulas are purely syntactical objects. They *don't have any meaning* – to stress this, let's keep all the brackets around (although as you get more and more comfortable with logic can probably start skipping some bracketing).[4]

Remark 2.1.3. Our formulas are **inductive objects**. That is, every formula is built by induction, using the steps (1)-(3), above. Thus, if we want to *prove* that some statement P is true of all formulas, it suffices to do the following:

- Base case: Prove that P is true of all propositional variables.
- Inductive step: Prove that if P is true of ϕ and ψ then P is true of $(\phi \wedge \psi)$, $(\phi \vee \psi)$, $(\phi \rightarrow \psi)$ and $(\neg\phi)$.

Another way of expressing the second bullet above is:

- Inductive step: Suppose that ϕ is of the form $(\phi_1 \wedge \phi_2)$ and P holds of ϕ_1 and of ϕ_2. Show that P holds of ϕ. Do the same for all other possible forms of ϕ.

Similarly, if we want to *define* some property S of formulas, we need to do the following:

- Base case: Define S for propositional variables.
- Inductive step: Assuming that S has been defined for ϕ and ψ, define S for $(\phi \wedge \psi)$, $(\phi \vee \psi)$, $(\phi \rightarrow \psi)$ and $(\neg\phi)$.

Let's illustrate this with an important example:

[4]There are binding conventions just like arithmetic. $\neg$ binds the strongest, then $\vee$ and $\wedge$ have the same binding power, and then $\rightarrow$ is the weakest. We still use brackets, when need be, to make things clear.

Definition 2.1.4. Let ϕ be a formula. We define the *set of subformulas* of ϕ, denoted $\mathsf{Sub}(\phi)$ inductively, as follows:

(1) For a propositional variable A, we define $\mathsf{Sub}(A) = \{A\}$

(2) Suppose that we have defined $\mathsf{Sub}(\phi_1)$ and $\mathsf{Sub}(\phi_2)$. Then:

- $\mathsf{Sub}(\phi_1 \wedge \phi_2) = \{\phi_1 \wedge \phi_2\} \cup \mathsf{Sub}(\phi_1) \cup \mathsf{Sub}(\phi_2)$,
- $\mathsf{Sub}(\phi_1 \vee \phi_2) = \{\phi_1 \vee \phi_2\} \cup \mathsf{Sub}(\phi_1) \cup \mathsf{Sub}(\phi_2)$,
- $\mathsf{Sub}(\phi_1 \to \phi_2) = \{\phi_1 \to \phi_2\} \cup \mathsf{Sub}(\phi_1) \cup \mathsf{Sub}(\phi_2)$, and
- $\mathsf{Sub}(\neg\phi_1) = \{\neg\phi_1\} \cup \mathsf{Sub}(\phi_1)$.

So we have now defined the set of subformulas for any formula ϕ. We say that ψ is a *proper subformula* of ϕ is $\psi \in \mathsf{Sub}(\phi) \setminus \{\phi\}$.

This may seem complicated, but after just one example it will make all the sense in the world:

Example 2.1.5. Let ϕ be the following formula $((A_1 \to A_2) \vee (A_2 \to \neg A_3))$. To compute

$$\mathsf{Sub}(\phi) := \mathsf{Sub}(((A_1 \to A_2) \vee (A_2 \to \neg A_3)))$$

we go by the definition:

- Step 1. $\mathsf{Sub}(((A_1 \to A_2) \vee (A_2 \to \neg A_3))) = \{(A_1 \to A_2) \vee (A_2 \to \neg A_3)\} \cup \mathsf{Sub}((A_1 \to A_2)) \cup \mathsf{Sub}((A_2 \to \neg A_3))$.
- Step 2. $\mathsf{Sub}((A_1 \to A_2)) = \{(A_1 \to A_2)\} \cup \mathsf{Sub}(A_1) \cup \mathsf{Sub}(A_2)$, $\mathsf{Sub}((A_2 \to \neg A_3)) = \{(A_2 \to \neg A_3)\} \cup \mathsf{Sub}(A_2) \cup \mathsf{Sub}(\neg A_3)$
- Step 3. $\mathsf{Sub}(A_1) = \{A_1\}$, $\mathsf{Sub}(A_2) = \{A_2\}$, $\mathsf{Sub}(\neg A_3) = \{\neg A_3\} \cup \mathsf{Sub}(A_3)$
- Step 4. $\mathsf{Sub}(A_3) = \{A_3\}$,

So all in all:

$$\mathsf{Sub}(\phi) = \{A_1, A_2, A_3, (\neg A_3), (A_1 \to A_2), (A_2 \to (\neg A_3)), \phi\}.$$

That wasn't that hard, now, was it?

Definition 2.1.6. Let ϕ be a formula. We define the *set of variables of* ϕ, denoted $\mathsf{Var}(\phi)$, inductively, as follows:

(1) For a propositional variable A, we define $\mathsf{Var}(A) = \{A\}$.

(2) Suppose that we have defined $\mathsf{Var}(\phi_1)$ and $\mathsf{Var}(\phi_2)$. Then:

- $\mathsf{Var}(\phi_1 \wedge \phi_2) = \mathsf{Var}(\phi_1) \cup \mathsf{Var}(\phi_2)$,
- $\mathsf{Var}(\phi_1 \vee \phi_2) = \mathsf{Var}(\phi_1) \cup \mathsf{Var}(\phi_2)$,
- $\mathsf{Var}(\phi_1 \to \phi_2) = \mathsf{Var}(\phi_1) \cup \mathsf{Var}(\phi_2)$, and
- $\mathsf{Var}(\neg\phi_1) = \mathsf{Var}(\phi_1)$.

In this way, we have defined $\mathsf{Var}(\phi)$ for all formulas ϕ. We call a formula ϕ a *propositional sentence* if $\mathsf{Var}(\phi) = \emptyset$.

Example 2.1.7. To find $\mathsf{Var}(((\neg(A \wedge B)) \to ((\neg C) \wedge (\neg B))))$ we go by the definition:

- Step 1. $\mathsf{Var}(((\neg(A \wedge B)) \to ((\neg C) \wedge (\neg B)))) = \mathsf{Var}((\neg(A \wedge B))) \cup \mathsf{Var}(((\neg C) \wedge (\neg B)))$.
- Step 2. $\mathsf{Var}((\neg(A \wedge B))) = \mathsf{Var}((A \wedge B))$, and $\mathsf{Var}(((\neg C) \wedge (\neg B))) = \mathsf{Var}((\neg(C))) \cup \mathsf{Var}((\neg B))$.
- Step 3. $\mathsf{Var}((A \wedge B)) = \mathsf{Var}(A) \cup \mathsf{Var}(B)$, $\mathsf{Var}((\neg C)) = \mathsf{Var}(C)$, $\mathsf{Var}(\neg(B)) = \mathsf{Var}(B)$.
- Step 4. $\mathsf{Var}(A) = \{A\}$, $\mathsf{Var}(B) = \{B\}$, $\mathsf{Var}(C) = \{C\}$ and $\mathsf{Var}(B) = \{B\}$.
- Step 5. $\mathsf{Var}(((\neg(A \wedge B)) \to ((\neg C) \wedge (\neg B)))) = \{A\} \cup \{B\} \cup \{C\} \cup \{B\} = \{A, B, C\}$

Okay, that was a pain, let's never do this again.

Exercise 2.1.8. Let ϕ be a formula. Show that $\mathsf{Var}(\phi) = \mathsf{Sub}(\phi) \cap \mathsf{Var}$ (where, remember, Var is the set of *all* propositional variables).

Definition 2.1.9. Let ϕ, ψ be formulas and A a propositional variable. The *substitution of ψ for A in ϕ*, denoted $\phi[\psi/A]$ is defined as follows:

(1) For the propositional variable B we define $B[\psi/A]$ as follows:

$$B[\psi/A] := \begin{cases} B & \text{if } A \neq B \\ \psi & \text{if } A = B. \end{cases}$$

(2) Suppose that we have defined $\phi_1[\psi/A]$ and $\phi_2[\psi/A]$. Then:

- $(\phi_1 \wedge \phi_2)[\psi/A] = (\phi_1[\psi/A]) \wedge \phi_2[\psi/A])$,
- $(\phi_1 \vee \phi_2)[\psi/A] = (\phi_1[\psi/A]) \vee \phi_2[\psi/A])$,

- $(\phi_1 \to \phi_2)[\psi/A] = (\phi_1[\psi/A]) \to \phi_2[\psi/A])$, and
- $(\neg\phi_2)[\psi/A] = (\neg\phi_1[\psi/A])$.

In this way, we have defined $\phi[\psi/A]$ for all formulas ϕ.

The following remark is also by induction, but a different kind of induction:

Remark 2.1.10. We have defined $\phi[\psi/A]$ and we iterate the definition above. Indeed, Suppose that $\psi_1, \ldots, \psi_{n+1}$ are formulas and $A_1, \ldots, A_{n+1}$ are propositional variables, and $\phi[\psi_1/A_1, \ldots, \psi_n/A_n]$ has been defined. Then:

$$\phi[\psi_1/A_1, \ldots, \psi_{n+1}/A_{n+1}] = (\phi[\psi_1/A_1, \ldots, \psi_n/A_n,])[\psi_{n+1}/A_{n+1}].$$

Let's see how to use this stuff in practice:

Lemma 2.1.11. *Suppose that $A \notin \mathsf{Var}(\phi)$. Then, $\phi[\psi/A] = \phi$ for any formula ψ.*

This is of course a somewhat silly lemma, but it'll help us understand better how to prove things by **induction on the structure of formulas**, i.e. as in Remark 2.1.3

PROOF. By induction.

- Base case: If ϕ is a propositional variable and $A \notin \mathsf{Var}(A)$, then $\phi = B$, for some propositional variable $B \neq A$. Thus $\phi[\psi/A] = B[\psi/A] = B = \phi$.
- Inductive step. Suppose that the theorem holds of ϕ_1 and ϕ_2 then we have to show that it holds of $(\phi_1 \wedge \phi_2), \ldots$. I'll only do one case here:
  (1) By inductive hypothesis, we know that "If $A \notin \mathsf{Var}(\phi_i)$ then $\phi_i[\psi/A] = \phi_i$, for $i = 1, 2$. Suppose that ϕ is of the form $(\phi_1 \wedge \phi_2)$ and that $A \notin \mathsf{Var}(\phi)$. Then, $A \notin \mathsf{Var}(\phi_1) \cup \mathsf{Var}(\phi_2)$, by definition of Var for conjunctions. Then $\phi_1[\psi/A] = \phi_1$ and $\phi_2[\psi/A] = \phi_2$, by inductive hypothesis. Thus:
  $$\begin{aligned}\phi[\psi/A] &= (\phi_1 \wedge \phi_2)[\psi/A] \\ &= (\phi_1[\psi/A] \wedge \phi_2[\psi/A]) \\ &= (\phi_1 \wedge \phi_2) \\ &= \phi.\end{aligned}$$
  (2) The rest of the cases are left as exercises.

□

Exercise 2.1.12. Complete the proof of the lemma above.

2.2. Truth tables. Okay, let's give meaning to our formulas. First of all, every propositional variable can be either true or false (if you're not with me on that one, you may be a lost cause), and for each of these scenarios we need to figure out if our formula is true or false. To make sense of this, we will use the following symbols to denote *truth* and *falsity*:

$$T \text{ means Truth, and } F \text{ means Falsity.}$$

The logical connectives we defined are **truth functions**, their inputs come from $\{F, T\}$ (to some power) and their output is either T or F.

So, to define the meaning of a formula we should first define what kind of functions our constant symbols and our logical connectives are, and take it from there.

Let X be a set. We say that a function is **unary** on X if its domain is X (i.e. it takes as input a single element of X) and **binary** on X if its domain is $X \times X$ (i.e. it takes as input two elements of X).[5] For example, the function $f : \mathbb{Z} \to \mathbb{Z}$ taking as input any integer x and returning $-x$ is a unary function on $\mathbb{Z}$, while the function $g : \mathbb{Z} \times \mathbb{Z} \to \mathbb{Z}$ taking as input two integers x and y and returning $x + y$ is a binary function on $\mathbb{Z}$.

In this chapter functions will have as domain the set $\{T, F\}$ of truth values. We will define truth functions via **truth tables**.

(1) NEGATION: The truth function $\neg$ defines is a unary function. That is, $f_\neg : \{\bot, \top\} \to \{\bot, \top\}$. Our intuition is that the negation truth should be falsity and the negation of falsity should be truth. This is expressed in the following truth table:

x	$f_\neg(x)$
F	T
T	F

What this means is that when x has the value T, $f_\neg(x)$ has the value F and when x has the value F, $f_\neg(x)$ has the value T.

[5] This terminology will be generalised more later down the line, but we'll stick to these terms for now.

(2) CONJUNCTION: The truth function $\wedge$ defines is a binary function. That is, $f_\wedge : \{F, T\} \times \{F, T\} \to \{F, T\}$. Our intuition is that the conjunction of two statements should be true precisely when they both are true:

x	y	$f_\wedge(x, y)$
F	F	F
F	T	F
T	F	F
T	T	T

So, as we'd expect $f_\wedge(x, y)$ is true when x and y are both true.

(3) DISJUNCTION: The truth function $\vee$ defines is also a binary function. That is, $f_\vee : \{F, T\} \times \{F, T\} \to \{F, T\}$. It may not be exactly obvious how $\vee$ should behave (in fact in the real world, there are two ways in which "or" is used: the *exclusive way* and the *inclusive way*). In logic, we're inclusive people and we will *always* adopt the latter. Thus, the disjunction of two statements should be true precisely when one of them is true:

x	y	$f_\vee(x, y)$
F	F	F
F	T	T
T	F	T
T	T	T

This is where the easy things end. The following discussion is mainly here to give intuition, but it may not be too intuitive, if you read it and it confuses you, don't overthink it.

Informal-ish Discussion. Let's briefly look back at our set theory. We defined the intersection of two sets to be the set that contains the elements that appear in both sets. Let X and Y be sets, and A and B be the statements $x \in X$ and $x \in Y$, respectively. Then $x \in X \cap Y$ if and only if $A \wedge B$ holds. Similarly, $x \in X \cup Y$ if and only if $A \vee B$ holds. To define implication, a good way of thinking about it is in terms of subsets. We want to say that $X \subseteq Y$ if and only if $A \to B$. What this means is the following:

- If $x \in X$ then the only valid outcome for $X \subseteq Y$ is that $x \in Y$, i.e. if A it true, then B should be true, in order for $A \to B$ to hold.

- If $x \notin X$ then both $x \in Y$ or $x \notin Y$ are valid outcomes for $X \subseteq Y$, i.e. for $A \to B$ to be true, so $A \to B$ is true when A is false and B is either true or false.
- If $x \notin Y$ and $x \in X$ then X is not a subset of Y. So if A is true and B is false, then $A \to B$ is false.

Sorry if that was confusing. Let's summarise –if you skipped the above, below is the definition of $\to$:

(3) IMPLICATION: The truth function $\to$ defines is also a binary function. That is, $f_\to : \{F, T\} \times \{F, T\} \to \{F, T\}$. The truth table of this function is given below:

x	y	$f_\to$
F	F	T
F	T	F
T	F	T
T	T	T

Now that we have some truth tables under our belt, we can start connecting the suckers. We will be doing this in a pretty natural way. Indeed, in the truth tables above, x and y could be propositional variables whose truth value we had already decided, but they could well have been formulas, in their own right (again whose truth value he had already defined). This is starting to smell like an inductive definition...

Going back to the reason we are doing all of this stuff in the first place, we want to abstract away arguments, right? And in our arguments, propositional variables are "placeholders" for English statements. And these English statements could, in a given situation, either be true or false. The next definition captures the idea of a "situation":

Definition 2.2.1. A *truth assignment* (or just *assignment*) is a function $\mathcal{A}$ which associates to every propositional variable a truth value, i.e. a function $\mathcal{A} : \mathsf{Var} \to \{T, F\}$. We will write $\mathbb{A}$ for the set of all assignments.

We can already define what the truth value of a propositional sentence is:

Definition 2.2.2. Let ϕ be a propositional formula and $\mathcal{A}$ an assignment. Then, the *valuation of ϕ under $\mathcal{A}$*, denoted $\mathsf{val}_\mathcal{A}(\phi)$ is defined as follows:

(1) $\mathsf{val}_\mathcal{A}(A) = \mathcal{A}(A)$, for any $A \in \mathsf{Var}$.

(2) Let ϕ_1 and ϕ_2 be propositional sentences and assume that $\mathsf{val}_{\mathcal{A}}(\phi_1)$ and $\mathsf{val}_{\mathcal{A}}(\phi_2)$ have been defined. Then:

   (a) $\mathsf{val}_{\mathcal{A}}(\phi_1 \wedge \phi_2) = f_{\wedge}(\mathsf{val}_{\mathcal{A}}(\phi_1), \mathsf{val}_{\mathcal{A}}(\phi_2))$.

   (b) $\mathsf{val}(\phi_1 \vee \phi_2) = f_{\vee}(\mathsf{val}_{\mathcal{A}}(\phi_1), \mathsf{val}_{\mathcal{A}}(\phi_2))$.

   (c) $\mathsf{val}(\phi_1 \to \phi_2) = f_{\to}(\mathsf{val}_{\mathcal{A}}(\phi_1), \mathsf{val}_{\mathcal{A}}(\phi_2))$.

   (d) $\mathsf{val}(\neg\phi_1) = f_{\neg}(\mathsf{val}_{\mathcal{A}}(\phi_1))$,

   where the functions in (a)-(d) are as we defined previously.

We will also write $\phi[\mathcal{A}]$ to denote $\mathsf{val}_{\mathcal{A}}(\phi)$, and we will write $\mathcal{A} \vDash \phi$ to denote that $\phi[\mathcal{A}] = T$ (and similarly $\mathcal{A} \nvDash \phi$ to denote that $\phi[\mathcal{A}] = F$).

Let's illustrate this with a couple of examples:

Example 2.2.3. Suppose that $\mathcal{A}$ is an assignment such that $\mathcal{A} : A \mapsto T, B \mapsto F$ and $\mathcal{A}'$ an assignment such that $\mathcal{A}' : A \mapsto T$ and $\mathcal{A}' : B \mapsto T$. Then:

$$\mathcal{A} \nvDash (A \to B), \text{ but } \mathcal{A}' \vDash (A \to B).$$

Intuitively, we want to think of formulas as being themselves truth functions. The unfortunate reality of life is, though, that they are not. In fact, what they are is functions from $\mathbb{A}$, the set of all assignments, to $\{T, F\}$.

In the examples above, to see if an assignment makes a formula true, we ONLY had to look at the variables that appear in the formula. This is very much a general fact:

THEOREM 2.2.4. *Let ϕ be a formula and $\mathcal{A}, \mathcal{B} \in \mathbb{A}$ two assignments. Suppose that for all $A \in \mathsf{Var}(\phi)$ we have that $\mathcal{A}(A) = \mathcal{B}(A)$. Then $\mathcal{A} \vDash \phi$ if and only if $\mathcal{B} \vDash \phi$.*

PROOF. By induction on the structure of ϕ, see next exercise. □

Exercise 2.2.5. Write out the details of the proof of Theorem 2.2.4

The point of the theorem above is that the truth function of a formula depends just on the propositional variables that appear in it. Since formulas are finite objects, they only contain finitely many propositional variables, and thus, to fully describe what ϕ does we only need to specify the values it takes on the possible assignments of its variables.

A little more frivolous notation before we get back to our good old friends the truth tables

Notation 2.2.6. Let $g, h : \mathbb{A} \to \{T, F\}$ be functions. We define:

(1) $g \wedge h : \mathbb{A} \to \{T, F\}$ to be the function
$$\begin{aligned} g \wedge h : \mathbb{A} &\to \{T, F\} \\ \mathcal{A} &\mapsto f_{\wedge}(g(\mathcal{A}), h(\mathcal{A})). \end{aligned}$$

(2) $g \vee h : \mathbb{A} \to \{T, F\}$ to be the function
$$\begin{aligned} g \vee h : \mathbb{A} &\to \{T, F\} \\ \mathcal{A} &\mapsto f_{\vee}(g(\mathcal{A}), h(\mathcal{A})). \end{aligned}$$

(3) $g \to h : \mathbb{A} \to \{T, F\}$ to be the function
$$\begin{aligned} g \to h : \mathbb{A} &\to \{T, F\} \\ \mathcal{A} &\mapsto f_{\to}(g(\mathcal{A}) \to h(\mathcal{A})). \end{aligned}$$

(4) $\neg g : \mathbb{A} \to \{T, F\}$ to be the function
$$\begin{aligned} \neg g : \mathbb{A} &\to \{T, F\} \\ \mathcal{A} &\mapsto f_{\neg}(g(\mathcal{A})). \end{aligned}$$

After all of this, we have essentially achieved triviality! Indeed:

THEOREM 2.2.7. *Let ϕ_1, ϕ_2 be formulas. Then:*

(1) $f_{\phi_1 \wedge \phi_2} = f_{\phi_1} \wedge f_{\phi_2}$

(2) $f_{\phi_1 \vee \phi_2} = f_{\phi_1} \vee f_{\phi_2}$

(3) $f_{\phi_1 \to \phi_2} = f_{\phi_1} \to f_{\phi_2}$

(4) $f_{\neg \phi_1} = \neg f_{\phi_1}$

PROOF. We only prove (1). The rest are left as exercises. We have to show that for all $\mathcal{A} \in \mathbb{A}$ we have that $f_{\phi_1 \wedge \phi_2}(\mathcal{A}) = f_{\phi_1}(\mathcal{A}) \wedge f_{\phi_2}(\mathcal{A})$. To this end, let $\mathcal{A} \in \mathbb{A}$. Then we have that:

$$\begin{aligned} f_{\phi_1 \wedge \phi_2}(\mathcal{A}) &= (\phi_1 \wedge \phi_2)[\mathcal{A}] \\ &= \mathsf{val}\left((\phi_1 \wedge \phi_2)[\mathcal{S}_{\mathcal{A}}]\right) \\ &= f_{\wedge}(\mathsf{val}(\phi_1[\mathcal{S}_{\mathcal{A}}]), \mathsf{val}(\phi_2[\mathcal{S}_{\mathcal{A}}])) \\ &= \phi_1[\mathcal{A}] \wedge \phi_2[\mathcal{A}] \\ &= f_{\phi_1}(\mathcal{A}) \wedge f_{\phi_2}(\mathcal{A}). \end{aligned}$$

□

Exercise 2.2.8. Show (2)-(4) in the theorem above.

Now that you've all hopefully forgiven me for my wild amounts of pedantry, I'm happy to announce that all of this becomes much easier with truth tables. Indeed, if $A_1, \ldots, A_n$ is a finite set of propositional variables, then we can write down all assignments restricted to $A_1, \ldots, A_n$ as a table of height 2^n. For example, all possible assignments of A_1, A_2, A_3 can be written down as the following table:

A_1	A_2	A_3
F	F	F
F	F	T
F	T	F
F	T	T
T	F	F
T	F	T
T	T	F
T	T	T

Similarly, all assignments of A_1, A_2, A_3, A_4 can be written down as the following table:

A_1	A_2	A_3	A_4
F	F	F	F
F	F	F	T
F	F	T	F
F	F	T	T
F	T	F	F
F	T	F	T
F	T	T	F
F	T	T	T
T	F	F	F
T	F	F	T
T	F	T	F
T	F	T	T
T	T	F	F
T	T	F	T
T	T	T	F
T	T	T	T

Hopefully, by this point, you get the point. Now, by Theorem 2.2.7, given a complicated formula we just need to evaluate its truth value one connective at a time. For example, consider the formula ϕ given by:

$$((A_1 \to A_2) \vee (A_2 \to \neg A_3))$$

We can write down f_ϕ as a long truth table, as follows:

A_1	A_2	A_3	$\neg A_3$	$(A_1 \to A_2)$	$(A_2 \to \neg A_3)$	ϕ
F	F	F	T	T	T	T
F	F	T	F	T	T	T
F	T	F	T	T	T	T
F	T	T	F	T	F	T
T	F	F	T	F	T	T
T	F	T	F	F	T	T
T	T	F	T	T	T	T
T	T	T	F	T	F	F

Exercise 2.2.9. Write out the truth tables of the following formulas:

(1) $(A \to (B \to A)$.

(2) $((A \to (B \to C)) \to ((A \to B) \to (A \to C)))$.

(3) $(A \to (\neg A))$.

(4) $(\neg(A \vee B) \to ((\neg A) \wedge (\neg B)))$.

(5) $((\neg A) \wedge (\neg B))) \to (\neg(A \vee B))$

(6) $(((\neg A) \to (\neg B)) \to (((\neg B) \to A) \to A))$

Another important consequence of Theorem 2.2.4 is that we can define new truth functions, by specifying the values they take only at the assignments of the propositional variables that occur in them. That is, we can specify new truth functions by writing down their truth tables. Truth functions, naturally give rise to new connectives. Let's illustrate this via an example:

Example 2.2.10. We define a new binary truth function f as follows:

x	y	$f(x,y)$
F	F	T
F	T	F
T	F	F
T	T	T

Define a new connective $\leftrightarrow$ by setting, for each assignment $\mathcal{A} \in \mathbb{A}$:

$$(\phi_1 \leftrightarrow \phi_2)[\mathcal{A}] := f(\phi_1[\mathcal{A}], \phi_2[\mathcal{A}]).$$

That is to say, for any assignment $\mathcal{A} \in \mathbb{A}$ we have that:

$$(\phi_1 \leftrightarrow \phi_2)[\mathcal{A}] = f(\phi_1[\mathcal{A}], \phi_2[\mathcal{A}]) = \begin{cases} T & \text{if } \phi_1[\mathcal{A}] = \phi_2[\mathcal{A}] \\ F & \text{if } \phi_1[\mathcal{A}] \neq \phi_2[\mathcal{A}] \end{cases}$$

In this sense, any function $f : \{T, F\}^n \to \{T, F\}$ defines a new connective (not all of them deserve their own symbols though). We will see in a minute that actually using only the functions $f_\wedge$, $f_\vee$, and $f_\neg$ we can already define every function $f : \{T, F\}^n \to \{T, F\}$.

2.3. Tautologies are always true.

Definition 2.3.1. We say that a formula ϕ is a *tautology*, if for all $\mathcal{A} \in \mathbb{A}$ we have that $\mathcal{A} \vDash \phi$. In this case, we write $\vDash \phi$.

To prove that a formula is a tautology, we just need to write down its truth table and show that every entry in the final column is T.

Example 2.3.2. The formula $(A \vee (\neg A))$ is a tautology. Indeed:

A	$\neg A$	$A \vee (\neg A)$
T	F	T
F	T	T

This tautology is important enough to have a name, it's called **the law of excluded middle**. Some people don't think this is a tautology.

Here's another tautology:

Example 2.3.3. The formula $((A \wedge (A \to B)) \to B)$ is a tautology. Indeed:

A	B	$(A \to B)$	$(A \wedge (A \to B)$	$((A \wedge (A \to B)) \to B)$
F	F	T	F	T
F	T	T	F	T
T	F	F	F	T
T	T	T	T	T

This tautology is **modus ponens**. Everyone agrees on that one.

Exercise 2.3.4. Prove that the following formulas are tautologies:

(1) $(A \to (B \to A))$

(2) $((A \to (B \to C)) \to ((A \to B) \to (A \to C)))$

(3) $(\neg A \to \neg B) \to ((\neg A \to B) \to A))$

Definition 2.3.5. We say that a formula ϕ **logically entails** a formula ψ if, for every $\mathcal{A} \in \mathbb{A}$ such that $\phi[\mathcal{A}] = T$ we have that $\psi[\mathcal{A}] = T$. In this case, we write $\phi \vDash \psi$. More generally, if $\Gamma = \{\phi_1, \dots, \phi_n, \dots\}$ is a set of formulas, we say that Γ **logically entails** a formula ψ if, for every $\mathcal{A} \in \mathbb{A}$ such that $\phi_i[\mathcal{A}] = T$ for all $\phi_i \in \Gamma$ (at the same time) we have that $\psi[\mathcal{A}] = T$. In this case, we write $\Gamma \vDash \psi$.

Example 2.3.6. The formula $A \wedge B$ logically entails A. The formula $A \wedge (A \to B)$ logically entails B. The formula $(A \wedge \neg A)$ logically entails EVERY formula (just like the empty set is a subset of every set, there are no assignments making $(A \wedge \neg A)$ true,[6] hence every such makes any formula true).

Definition 2.3.7. We say that a formula ϕ is *logically equivalent* to a formula ψ if $\phi \leftrightarrow \psi$ is a tautology.

Let's start putting definitions together:

Lemma 2.3.8. *Let ϕ and ψ be formulas. Then, the following are equivalent:*

(1) ϕ logically implies ψ.

(2) $(\phi \to \psi)$ is a tautology.

In symbols, $\phi \vDash \psi$ if and only if $\vDash (\phi \to \psi)$.

[6] Check this!

PROOF. Assume (1) and let $\mathcal{A} \in \mathbb{A}$. If $\mathcal{A} \vDash \phi$, then by definition $\mathcal{A} \vDash \psi$, and hence $\mathcal{A} \vDash \phi \to \psi$. If on the other hand, $\mathcal{A} \nvDash \phi$, then $\mathcal{A} \vDash \phi \to \psi$. [Why??]

Conversely, assume (2). Let $\mathcal{A}$ be an assignment. If $\mathcal{A} \vDash \phi$ then since $\mathcal{A} \vDash \phi \to \psi$ we have that $\mathcal{A} \vDash \psi$. □

Remark 2.3.9. Truth tables are an effective (i.e. you can sit down and do it) way of checking if a formula is a tautology. Hence, we also have an effective way of checking if a formula logically implies another.

Just to make our world salad a bit more spicy, we also have the following definition:

Definition 2.3.10. A formula ϕ is called *unsatisfiable* (or *contradictory*) if for all $\mathcal{A} \in \mathbb{A}$ we have that $\phi[\mathcal{A}] = F$. We say that a set of formulas Γ is *unsatisfiable* if for all $\mathcal{A} \in \mathbb{A}$ there is some $\phi \in \Gamma$ such that $\phi[\mathcal{A}] = F$.

Clearly, ϕ is contradictory if and only if $(\neg\phi)$ is a tautology. So far so good, let's start proving some tautologies:

Lemma 2.3.11. *Let Γ be a set of formulas and ϕ, ψ formulas. If Γ logically entails $(\phi \to \psi)$ and ϕ, then Γ logically entails ψ. In particular, if ϕ and $\phi \to \psi$ are tautologies, then so is ψ.*

In symbols, if $\Gamma \vDash \phi$ and $\Gamma \vDash (\phi \to \psi)$ then $\Gamma \vDash \psi$.

PROOF. Suppose that $\Gamma \vDash \phi \to \psi$ and $\Gamma \vDash \phi$. Let $\mathcal{A}$ be an assignment such that $\mathcal{A} \vDash \Gamma$. If $\psi[\mathcal{A}] = F$, since $\phi \to \psi[\mathcal{A}] = T$, we must have that $\phi[\mathcal{A}] = F$. [Why??] But by assumption, we have that $\phi[\mathcal{A}] = T$, which is a contradiction. Thus $\mathcal{A} \vDash \psi$, as required. □

Proposition 2.3.12. *Suppose that ϕ is a tautology. Let $\psi_1, \ldots, \psi_n$ be formulas and $A_1, \ldots, A_n$ be distinct propositional variables. Then $\phi[\psi_1/A_1, \ldots, \psi_n/A_n]$ is a tautology.*

PROOF. It suffices to show this for $n = 1$ [Why?] Suppose that ϕ is a tautology. Let $\mathcal{A}$ be an assignment. We need to show that $\mathcal{A} \vDash \phi[\psi_1/A_1]$. Define a new assignment $\mathcal{A}'$ by setting:

$$\mathcal{A}' : A \mapsto \begin{cases} \mathsf{val}_{\mathcal{A}}(\psi_1) & \text{if } A = A_1 \\ \mathcal{A}(A) & \text{otherwise.} \end{cases}$$

Then, we obviously have that:

$$\mathcal{A} \vDash \phi[\psi_1/A_1] \text{ if and only if } \mathcal{A}' \vDash \phi.$$

[Formally, we should prove the above by induction on the structure of formulas.] But, since ϕ is a tautology, we have that $\mathcal{A}' \vDash \phi$, so $\mathcal{A} \vDash \phi[\psi_1/A_1]$, and hence $\phi[\psi_1/A_1]$ is a tautology. □

The next proposition will be very useful. It will read a bit confusing at first, but it's actually just a lot of words to say something very simple. The moral is that if you replace things by things that are the same as they were then what you started with and what you ended with are the same. So, up to logical equivalence, Theseus's ship is a solved problem.

Proposition 2.3.13. *If ϕ be a formula, $\psi \in \mathsf{Sub}(\phi)$, and suppose that ϕ' is obtained by replacing each occurence of ψ in ϕ by the formula χ. If ψ and χ are logically equivalent, then so are ϕ and ϕ'.*

PROOF. We will show that $(\psi \leftrightarrow \chi) \rightarrow (\phi \leftrightarrow \phi')$ is a tautology (where $\leftrightarrow$ is the binary connective defined in Example 2.2.10). Let $\mathcal{A}$ be any assignment. If $\mathsf{val}_{\mathcal{A}}(\psi) \neq \mathsf{val}_{\mathcal{A}}(\chi)$, then $\mathcal{A} \vDash (\psi \leftrightarrow \chi) \rightarrow (\phi \leftrightarrow \phi')$ [Why?]. If, on the other hand, $\mathsf{val}_{\mathcal{A}}(\psi) = \mathsf{val}_{\mathcal{A}}(\chi)$, then again $\mathcal{A} \vDash (\psi \leftrightarrow \chi) \rightarrow (\phi \leftrightarrow \phi')$ [This should be clear, since ϕ and ϕ' only differ in that where one contains ψ the other contains χ and $\mathsf{val}_{\mathcal{A}}(\psi) = \mathsf{val}_{\mathcal{A}}(\phi)$ – again formally this should be proved by induction]. The result follows. □

Parts of the following exercise will sometimes make our lives easier in the future.

Exercise 2.3.14. Prove the following logical equivalences:

(1) $(A \wedge B) \wedge C$ and $A \wedge (B \wedge C)$

(2) $(A \vee B) \vee C$ and $A \vee (B \vee C)$.

(3) $A \rightarrow (B \rightarrow C)$ and $(A \wedge B) \rightarrow C$.

(4) $A \wedge (B \vee C)$ and $(A \wedge B) \vee (A \wedge B)$.

(5) $A \vee (B \wedge C)$ and $(A \vee B) \wedge (A \vee B)$.

(6) $(A \wedge B) \vee \neg B$ and $A \vee \neg B$.

(7) $(A \vee B) \wedge \neg B$ and $A \wedge \neg B$.

(8) $A \rightarrow B$ and $\neg B \rightarrow \neg A$.

(9) $A \leftrightarrow B$ and $B \leftrightarrow A$.

(10) $(A \leftrightarrow B) \leftrightarrow C$ and $A \leftrightarrow (B \leftrightarrow C)$.

The first two parts of the previous exercise tell us that when dealing with multiple conjunctions (or disjunctions), the way we bracket them does not matter. This allows us to introduce the following notation:

Notation 2.3.15. We write $\bigwedge_{i=1}^{n} \phi_i$ as shorthand for $\phi_1 \wedge (\phi_2 \wedge (\cdots \wedge (\phi_{n-1} \wedge \phi_n) \dots))$, and $\bigvee_{i=1}^{n} \phi_i$ as shorthand for $\phi_1 \vee (\phi_2 \vee (\cdots \vee (\phi_{n-1} \vee \phi_n) \dots))$.

2.4. Enough connectives are enough. The goal of this section is to show that *every* truth function can be expressed as a combination of the truth functions $f_\wedge, f_\vee$ and $f_\neg$. The upshot of this is that for every formula ϕ, there is a formula ψ built only by using the Boolean connectives $\wedge, \vee$ and $\neg$ which is logically equivalent to ϕ.

Things get a bit murky because on the one hand we have truth functions and on the other hand we have formulas, which are functions $\mathbb{A} \to \{T, F\}$.

Dirty-dirty formalism. The functions from $\mathbb{A} \to \{T, F\}$ which only depend on the assignment of a finite number of variables turn out to be the functions that we are interested in. Indeed, mathematical logic (the way we do it now) is only interested in finitary formulas.

Definition 2.4.1. A function $f : \mathbb{A} \to \{T, F\}$ is called a *truth function* if there exists a finite set of variables $\{A_1, \dots, A_n\} \subseteq \mathsf{Var}$ such that for all assignments $\mathcal{A}, \mathcal{B}$ we have that if:

$$\mathcal{A}_i \restriction_{\{A_1,\dots,A_n\}} = \mathcal{B} \restriction_{\{A_1,\dots,A_n\}}$$

Then, $f(\mathcal{A}) = f(\mathcal{B})$. In this case, we call $A_1, \dots, A_n$ the *support* of f, denoted $\mathsf{supp}(f)$.

Given a formula ϕ we can define a function $f_\phi : \mathbb{A} \to \{T, F\}$ in the obvious way:

$$\begin{aligned} f_\phi : \mathbb{A} &\to \{T, F\} \\ \mathcal{A} &\mapsto \mathsf{val}_{\mathcal{A}}(\phi). \end{aligned}$$

Thus, by Theorem 2.2.4, if ϕ is a formula, then its truth function f_ϕ is indeed a truth function. In fact, Theorem 2.2.4 shows that $\mathsf{supp}(f_\phi) = \mathsf{Var}(\phi)$.

Remark 2.4.2. Every propositional variable gives rise to a truth function, namely given a propositional variable $A \in \mathsf{Var}$, we can define the function $f_A : \mathbb{A} \to \{T, F\}$ by setting $f_A(\mathcal{A}) := \mathcal{A}(A)$. The support of f_A is just $\{A\}$.

You may be, at this point, wondering what the hell the point is.

Remark 2.4.3. The point is that truth functions can be fully described by a finite truth tables. Indeed, if f is a truth function, then for any assignment $\mathcal{A} \in \mathbb{A}$ the value of $f(\mathcal{A})$ is determined by $\mathcal{A} \restriction_{\mathsf{supp}(f)}$, thus, if we list all the $2^{|\mathsf{supp}(f)|}$ possible assignments of the elements of $\mathsf{supp}(f)$, the list of values that f takes on each assignment fully determines f on all $\mathbb{A}$.

We call a truth function whose support has size n an **n-ary truth function**. We call n the **arity** of f. Given any two n-ary truth functions, we can always assume that they have the same support. This follows immediately by the next lemma:

Lemma 2.4.4. *Let $f : \mathbb{A} \to \{T, F\}$ be an n-ary truth function. Suppose that $\mathsf{supp}(f) = \{A_1, \ldots, A_n\}$. Then, for any set $\{B_1, \ldots, B_n\} \subseteq \mathsf{Var}$, there is a truth function $f' : \mathbb{A} \to \{T, F\}$ whose support is $\{B_1, \ldots, B_n\}$ and such that for all $\mathcal{A}, \mathcal{B} \in \mathbb{A}$, if $\mathcal{A}(A_i) = \mathcal{B}(B_i)$ for all $i \leq n$, then $f(\mathcal{A}) = f'(\mathcal{B})$.*

PROOF. Define f' to be the following truth function:

$$\begin{aligned} f' : \mathbb{A} &\to \{T, F\} \\ \mathcal{A} &\mapsto f(\mathcal{B}), \end{aligned}$$

where $\mathcal{B}$ is the assignment $\mathcal{B}(B_i) := \mathcal{A}(A_i)$ for each $i \leq n$ and $\mathcal{B}(A) = F$, for all $A \in \mathsf{Var} \setminus \{B_1, \ldots, B_n\}$. It is easy to see that g is a truth function (i.e. that it has finite support). □

Fix a countable subset $\{A_1, A_2, \ldots\} \subseteq \mathsf{Var}$. By the lemma above, if f is an n-ary truth function, we can identify it with a truth function whose support is $A_1, \ldots, A_n$. For any assignment $\mathcal{A} \in \mathbb{A}$, f is only determined by $\mathcal{A} \restriction_{\{A_1, \ldots, A_n\}}$. Thus, by writing out the 2^n rows of the truth table of f restricted to its support, can view f as a true truth function $\{T, F\}^n \to \{T, F\}$. More formally, $\{T, F\}^n$ is just the set of n-tuples whose entries consist of T or F, and if f is an n-ary truth function, with support $\{A_1, \ldots, A_n\}$ then we define the function $\bar{f} : \{T, F\}^n \to \{T, F\}$ as follows:

$$\begin{aligned} \bar{f} : \{T, F\}^n &\to \{T, F\} \\ (x_1, \ldots, x_n) &\mapsto f(A_1 \mapsto x_1, \ldots, A_n \mapsto x_n), \end{aligned}$$

Notation 2.4.5. From now on, we will view identify each truth function f with the function $\bar{f} : \{T, F\}^n \to \{T, F\}$, where $n = |\mathsf{supp}(f)|$, that it induces.

Example 2.4.6. The truth function corresponding to a propositional variable is a **unary** (i.e. of arity 1) which returns true to input true and false to input false, i.e. it is the identity function. The only other unary function is the truth function of $\neg$. For our old connectives this is nothing new, indeed, it says that $f_{A\wedge B}$ is just $f_\wedge$ (the point being that $f_{A\wedge C}$ which is technically a different function is also identified with $f_\wedge$).

Definition 2.4.7. Let $\mathcal{F} := \{f_1 : \{T, F\}^{n_1} \to \{T, F\}, \ldots, f_k : \{T, F\}^{n_k} \to \{T, F\}\}$ be a set of truth functions. An *$\mathcal{F}$-term* is defined, inductively, as follows:

(1) Any propositional variable A_n is an $\mathcal{F}$-term.

(2) For each $f_i \in \mathcal{F}$, and $\mathcal{F}$-terms $t_1, \ldots, t_{n_i}$, $f_i(t_1, \ldots, t_{n_i})$ is an $\mathcal{F}$-term.

(3) All $\mathcal{F}$-terms are built like this.

Lemma 2.4.8. *$\mathcal{F} := \{f_1 : \{T, F\}^{n_1} \to \{T, F\}, \ldots, f_k : \{T, F\}^{n_k} \to \{T, F\}\}$ be a set of truth functions and h an $\mathcal{F}$-term. Then, h defines a truth function.*

Proof. The proof is by induction on the structure of $\mathcal{F}$-terms. The base case is trivial since all propositional variables define a truth function. Now, suppose that $f : \{T, F\}^n \to \{T, F\}$ is a truth function and $t_1, \ldots, t_n$ are $\mathcal{F}$-terms. By inductive hypothesis, for each $i \leq n$ we have that t_i is a truth function (of some arity) say r_i. Define:

$$h : \{T, F\}^m \to \{T, F\},$$

where $m = \sum_{i=1}^n r_i$ by setting $h(\bar{x}_1, \ldots, \bar{x}_n)$ to be $f(t_1(\bar{x}_1), \ldots, t_n(\bar{x}_n))$. To see that this is a truth function, we only need to observe that the support of h is precisely the union of the supports of the t_i's, so it is finite. □

This is another inductive definition, which is meant to capture the concept of "complicated function composition". Let's illustrate this by a couple of examples:

Example 2.4.9. Let $\mathcal{F}_1 = \{f_1 : \{T, F\}^2 \to \{T, F\}, f_2 : \{T, F\}^2 \to \{T, F\}, f_3 : \{T, F\} \to \{T, F\}\}$ be truth functions. We can build $\mathcal{F}$-terms which correspond by the previous lemma to new truth functions in various ways, for example, we can define:

$$\begin{aligned} h : \{T, F\}^3 &\to \{T, F\} \\ (x, y, z) &\mapsto f_1(f_2(x, y), z), \end{aligned}$$

for any $(x, y, z) \in \{T, F\}^3$. Similarly, we could also define:

$$\begin{aligned} h' : \{T, F\}^3 &\to \{T, F\} \\ (x, y, z) &\mapsto f_1(f_2(x, y), f_3(z)), \end{aligned}$$

for any $(x, y, z) \in \{T, F\}^3$ etc.

Example 2.4.10. Recall from Notation 2.2.6 that if g and h are functions $\mathbb{A} \to \{T, F\}$, then we write $g \wedge h$ for the truth function $f_\wedge(g, h)$. So, $g \wedge h$ is the truth function corresponding to the $\{f_\wedge, g, h\}$-term $f_\wedge(g(A), h(A))$, for any propositional variable A.

Exercise 2.4.11. Show that if $g_1, \ldots, g_n$ are truth functions, then the truth function $\bigwedge_{i=1}^n g_i$ (which is shorthand for $(g_1 \wedge (g_2 \wedge (\cdots \wedge (g_{n-1} \wedge g_n) \ldots))))$ is an $\{f_\wedge, g_1, \ldots, g_n\}$-term.

More generally, we say that a truth function $h : \{T, F\}^m \to \{T, F\}$ can be written as a **composition** of truth functions from $\mathcal{F} = \{f_i : \{T, F\}^{n_i} \to \{T, F\} : i \in I\}$ if it is the truth function corresponding to some $\mathcal{F}$-term.

We now come to our main definition of this subsection.

Definition 2.4.12. Let $\mathcal{F}$ be a set of truth functions. We say that $\mathcal{F}$ is *adequate* if for all $n \in \mathbb{N}$, every truth function $g : \{T, F\}^n \to \{T, F\}$ can be written as a composition functions from $\mathcal{F}$.

THEOREM 2.4.13. *The set $\{f_\wedge, f_\vee, f_\neg\}$ is adequate.*

PROOF. If f is a nullary truth function, then it is either always false, in which case it is $f_\bot$ or it is always true, in which case it is $f_\neg(f_\bot)$, where $\bot$ is shorthand for $A \wedge \neg A$, for any propositional variable A.

Let $f : \{T, F\}^n \to \{T, F\}$ be any n-ary truth function. We want to show that we can write f as a composition of $f_\wedge, f_\vee$ and $f_\neg$. By assumption, f is a function in variables $x_1, \ldots, x_n$, and it has 2^n possible inputs (the domain of f is $\{T, F\}^n$). We

list all the possible inputs to f by:

$$\begin{gathered} x_1^1 x_2^1 \dots x_n^1 \\ x_1^1 x_2^1 \dots x_n^1 \\ x_1^2 x_2^2 \dots x_n^2 \\ \vdots \\ x_1^m x_2^m \dots x_n^m \\ \vdots \\ x_1^{2^n} x_2^{2^n} \dots x_n^{2^n} \end{gathered}$$

For each $j \leq 2^n$, if $f(x_1^j, \dots, x_n^j) = T$, consider the following $\{f_\wedge, f_\neg\}$-term:

$$t_j(A_1, \dots, A_n) := g_1^j(A_1) \wedge g_2^j(A_2) \wedge \dots \wedge g_n^j(A_n),$$

where $g_i^j(A_i) = A_i$ if $x_i^j = T$ and $g_i^j(A_i) = f_\neg(A_i)$ if $x_i^j = F$.[7]

Now, let $\{j_1, \dots, j_k\} \subseteq 2^n$ be the set of all $j \leq 2^n$ such that $f(x_1^j, \dots, x_n^j) = T$. We claim that our desired function is the following:

$$h := t_{j_1}(A_1, \dots, A_n) \vee t_{j_2}(A_1, \dots, A_n) \vee \dots \vee t_{j_k}(A_1, \dots, A_n),$$

which by an adaptation of the previous exercise is an $\{f_\wedge, f_\vee, f_\neg\}$-term. It remains to show that for all $(x_1, \dots, x_n) \in \{T, F\}^n$ we have that $f(x_1, \dots, x_n) = h(x_1, \dots, x_n)$. But we understand the truth table of h. Indeed, since it is a big disjunction, of conjunctions we have that, given $(x_1, \dots, x_n) \in \{T, F\}^n$

$$\begin{aligned} h(x_1, \dots, x_n) = T &\text{ implies } t_{j_i}((x_1, \dots, x_n)) = T \text{ for some } i \leq 2^n \\ &\text{implies } g_1^{j_i}(x_1) \wedge g_2^{j_i}(x_2) \wedge \dots \wedge g_n^{j_i}(x_n) = T \\ &\text{implies } g_k^{j_i}(x_k) = T \text{ for all } k \leq n \\ &\text{implies } f(x_1, \dots, x_k) = T. \end{aligned}$$

Conversely, we have that

$$\begin{aligned} f(x_1, \dots, x_k) = T &\text{ implies } g_k^{j_i}(x_k) = T \text{ for all } k \leq n \text{ and some } i \leq 2^n \\ &\text{implies } h(x_1, \dots, x_n) \end{aligned}$$

and this concludes the proof. □

Okay, all of this has been really abstract. In part, we showed something rather cool. Any function from $\mathbb{A} \to \{T, F\}$ with finite support can be written as the truth

[7] We have here used the previous exercise to show that this is an $\{f_\wedge, f_\neg\}$-term.

function of a formula involving only three connectives, but the funky part is the first part: Any function from $\mathbb{A} \to \{T, F\}$ with finite support can be written as the truth function of a formula, so from now on, we are totally justified in taking formulas to be our primitive objects!

– End of digression –

Let's translate the previous theorem into a fact about formulas:

Corollary 2.4.14. *Let ϕ be a propositional formula. Then, there is a propositional formula ψ of the form:*

$$\bigvee_{i=1}^{l} \left(\bigwedge_{j=1}^{m} A_{i,j} \wedge \bigwedge_{j=1}^{n} \neg B_{i,j} \right),$$

where $A_{i,j}$ and $B_{i,j}$ are propositional variables, such that ϕ and ψ are logically equivalent.

Don't be too scared about the big symbols in the expression above! The corollary honestly just says that every formula is logically equivalent to a big disjunction of conjunctions of propositional or negated propositional variables!

The proof of the theorem (which you don't really have to have read) in disguise, gives us an actual factual algorithm for rewriting any formula (or actually any truth function) into one which is logically equivalent to it using only the connectives $\wedge, \vee$ and $\neg$:

- Step 1. Write out the truth table of the function/formula.
- Step 2. Forget about all the rows of the table that the function returns false and keep all the rows that it returns true.
- Step 3. For each row kept, look at the n variables involved and write down a long conjunction, where the i-th variable is negated if it shows up as false in the row and is kept as is, otherwise.
- Step 4. Take the disjunction of all the formulas you wrote down in Step 3.

Let's illustrate it with an actual example:

Example 2.4.15. Here's an old truth table:

A	B	$(A \to B)$
F	F	T
F	T	F
T	F	T
T	T	T

Step 1 done. For Step 2, we only keep rows 1,3, and 4. So

A	B	$(A \to B)$
F	F	T
F	T	F
T	F	T
T	T	T

So we write:

- First row: $\neg A \wedge \neg B$;
- Third row: $A \wedge \neg B$;
- Fourth row: $A \wedge B$.

All in all:

$$(\neg A \wedge \neg B) \vee (A \wedge \neg B) \vee (A \wedge B)$$

is our desired formula.

We say that formulas written as a disjunction of conjunctions of propositional or negated propositional variables are in **disjunctive normal form** (DNF). Practice makes perfect:

Example 2.4.16. Consider the formula $(A \to B) \to C$. Let's write this in DNF, following our algorithm:

First, we write out the truth table:

A	B	C	$A \to B$	$(A \to B) \to C$
F	F	F	T	F
F	F	T	T	T
F	T	F	T	F
F	T	T	T	T
T	F	F	F	T
T	F	T	F	T
T	T	F	T	F
T	T	T	T	T

Then, we focus on the rows that are true:

A	B	C	$A \to B$	$(A \to B) \to C$
F	F	F	T	F
F	F	T	T	T
F	T	F	T	F
F	T	T	T	T
T	F	F	F	T
T	F	T	F	T
T	T	F	T	F
T	T	T	T	T

Finally, for each row we write out a conjunction of the propositional variables either as they are (if they are true in the row) or negated (if they are false in the row):

$$\begin{aligned} &(\neg A \wedge \neg B \wedge C) && \text{Row 2} \\ \vee &(\neg A \wedge B \wedge C) && \text{Row 4} \\ \vee &(A \wedge \neg B \wedge \neg C) && \text{Row 5} \\ \vee &(A \wedge \neg B \wedge C) && \text{Row 6} \\ \vee &(A \wedge B \wedge C) && \text{Row 8} \end{aligned}$$

Now, to convince ourselves (if you haven't read the proof, this should be useful) that the algorithm works, let's do some checks:

A	B	C	$\neg A \wedge \neg B \wedge C$	$\neg A \wedge B \wedge C$	$A \wedge \neg B \wedge \neg C$	$A \wedge \neg B \wedge C$	$A \wedge B \wedge C$
F	F	F	F	F	F	F	F
F	F	T	T	F	F	F	F
F	T	F	F	F	F	F	F
F	T	T	F	T	F	F	F
T	F	F	F	F	T	F	F
T	F	T	F	F	F	T	F
T	T	F	F	F	F	F	F
T	T	T	F	F	F	F	T

Think about the colours in the table above for a bit; they should make a convincing case.

Now, as we've already seen, $A \wedge B$ is logically equivalent to $\neg(A \vee B)$. Hence, by Proposition 2.3.13 any formula using only $\wedge, \vee$ and $\neg$ is logically equivalent to a formula using only $\wedge$ and $\neg$. So we have more adequate sets of connectives, namely $\{\neg, \vee\}$, and similarly $\{\neg, \wedge\}$. Can we do any smaller?

THEOREM 2.4.17. *The set* $\{\rightarrow, \neg\}$ *is adequate.*

PROOF. We have that:

- $A \vee B$ is just $\neg A \rightarrow B$.
- $A \wedge B$ is just $\neg(\neg A \vee \neg B)$ which is just $\neg(A \rightarrow \neg B)$.

So, by Proposition 2.3.13 this set of connectives is adequate. □

That's all good and well, but could we use say a single connective?

Lemma 2.4.18. *Let* $\downarrow$ *be the binary connective (called* nor*) whose truth function is defined by:*

x	y	$f_{\downarrow}(x, y)$
F	F	T
F	T	F
T	F	F
T	T	F

Then $\{\downarrow\}$ *is adequate.*

PROOF. Observe that $\neg A$ is logically equivalent to $A \downarrow A$ and $A \to B$ is logically equivalent to $((A \downarrow A) \downarrow B) \downarrow ((A \downarrow A) \downarrow B)$ □

Exercise 2.4.19. Prove that the binary connective $|$ (called *nand*) whose truth function is defined by:

x	y	$f_{\mid}(x, y)$
F	F	T
F	T	T
T	F	T
T	T	F

is adequate.

Even though we won't spend too much time thinking about nor and nand, these guys are rather special:

FACT. *The only binary connectives that are adequate are* $\downarrow$ *and* $|$.

We're about to start going in the deeper end, so here's a couple of logic puzzles to get our spirits up:

Exercise 2.4.20 (From Mendelson's Introduction to Mathematical Logic)**.**

(1) A certain country is inhabited only by truthers (not that kind of truther's rather, people who always tell the truth) and liars (people who always lie). Moreover, the inhabitants will respond only to yes or no questions. A tourist comes to a fork in a road where one branch leads to the capital and the other leads down a cliff. There is no sign indicating which branch to take, but there is a native standing at the fork. What yes or no question should the tourist ask in order to determine which branch to take?

(2) In a different country, there are three kinds of people: workers (who of course always tell the truth), businessmen (who as we'd expect always lie), and students (who sometimes tell the truth and sometimes lie). At a fork in the road, one branch leads to the capital. A worker, a businessman and a student are standing at the side of the road but are not identifiable in any obvious way. By asking two yes or no questions, find out which fork leads to the capital (Each question may be addressed to any of the three.)