Mathematical Proof Methods for Logic*

Julian J. Schlöder

Winter 20/21

[*] I am indebted to my own teachers, Prof. Peter Koepke and Prof. Stefan Geschke, who taught me everything in these notes. Prof. Geschke's scriptum for *Einführung in die Logik und Modelltheorie* (Bonn, Summer 2010) provided an invaluable basis for the compilation of these notes.

Contents

1 What are mathematical proof methods for logic?

We want to study logic from an abstract, mathematical perspective. If you try to think about what logic *fundamentally is* and strip away everything concrete, you may arrive at something like this: if you have some sentences (or propositions) and some other sentence (or proposition), logic tells you whether these sentences *entail* that sentence. We want to make this idea very, very precise and then *mathematically prove* some things about different versions of entailment.

1.1 Preliminaries

To get things started we need some mathematical terminology.

Definition 1 (Set)**.** It is difficult to say what **sets** really *are*, but it is easy to say what they can *do*.

- Sets can contain things, for example $s = \{a, b, c\}$ is a set that contains a, b and c. We write $a \in s$ to say that a is a **member** of s.
  - But they don't have to contain things. There is a set not containing anything: the **empty set** $\emptyset$.
  - They can contain infinitely many things, like the set of all natural numbers.
  - Members of sets are not in any particular order. For example, $\{a, b, c\}$ and $\{c, b, a\}$ and $\{b, a, c\}$ are all the same set.
  - Multiply-mentioned members are simply redundant, for example $\{a, a, a, a, b, c\}$ is just a more complicated way of writing the set $\{a, b, c\}$.

  The idea is that the only thing that matters is *whether* something is a member, but not 'where' in the set it is or 'how often'.
- Sometimes, sets can be specified as containing all the things having some property P; in that case we write $\{x \mid P(x)\}$. For example, $\{x \mid x \text{ is an even number}\}$ is the set of even numbers.
- Given two sets s, s', one can form their *intersection*: $s \cap s'$ is the set containing all things that are both in s and in s'. Or, formally $s \cap s' = \{x \mid x \in s \text{ and } x \in s'\}$.
- Given two sets s, s', one can form their *union*: $s \cup s'$ is the set containing all things that are either in s or in s'. Or, formally $s \cup s' = \{x \mid x \in s \text{ or } x \in s'\}$.

We will often talk about sets and their *subsets* or *supersets*.

Definition 2 (Subset and Superset)**.** A set s is a **subset** of a set t (write: $s \subseteq t$) iff$_{\text{Def}}$ all members of s are also members of t. We say that s is a **proper subset** of t (write: $s \subset t$) iff$_{\text{Def}}$ $s \subseteq t$ and t has a member that is not also a member of s.

s is a **superset** of t (write: $s \supseteq t$) iff$_{\text{Def}}$ $t \subseteq s$. And s is a **proper superset** of t (write: $s \supset t$) iff$_{\text{Def}}$ $t \subset s$.

Sets are incredibly useful. But sometimes we want a little bit more structure.

Definition 3 (Ordered Sets)**.** An **ordered set** is like a set, except that the order and multiple mentions of members matters. For example $\langle a, b, c\rangle$ contains the things a, b and c in that order. The ordered sets

$\langle b, a, c\rangle$, $\langle a, a, b, c\rangle$ or $\langle a, b, c, a\rangle$ are all different from $\langle a, b, c\rangle$ and each other.

We call ordered sets **ordered pairs** if they have exactly two members, **triples** if they contain exactly three members, **quadruples** if they contain exactly four members… and although we could continue with **quint-**, **sept-**, **oct-** and **non-** we give up and talk about 5-tuples, 6-tuples, 7-tuples and so on. (In the literature, the term 'tuple' can occur as a synonym for 'ordered pair' but also as a synonym for 'ordered set'; we will just not use this term.)

There is a very useful operation we can define on ordered sets: concatenation.

Definition 4 (Concatenation)**.** Given two ordered sets s and t, their **concatenation** $s^\frown t$ is the ordered set that contains all members of s (in their order) and then all members of t (in their order) and nothing else.

For example, $\langle a, b, c\rangle^\frown\langle c, c, b, a\rangle = \langle a, b, c, c, c, b, a\rangle$.

With this we can define the formal concept of a *relation*. You are familiar with many relations: 'is equal' or 'smaller than' are relations on the numbers. Using sets, we can be *very precise* about what relations are.

Definition 5 (Relation)**.** A **relation** over a set A is a set of ordered pairs of members of A.

For example, the 'smaller than' relation can be written as the following:

$$s = \{\langle x, y\rangle \mid y - x \text{ is positive}\}$$

You can verify as an exercise that $x < y$ is the case if and only if $\langle x, y\rangle \in s$.

1.2 Consequence Relations

With the preliminaries in place, we can state **precisely** what kind of abstract objects we are interested in.

Definition 6 (Consequence Relation)**.** A **consequence relation** is a relation between sets of sentences [the **premisses**] and sentences [the **conclusion**].

If C is a consequence relation, it is convenient to write $\Gamma \therefore^C \varphi$ instead of $\langle \Gamma, \varphi\rangle \in C$.

A great many things are consequence relations.

- The **empty relation** is a consequence relation. Define C^{empty} to be the relation between sets of sentences and sentences so that *no sentence is ever related to any set of sentences.* Or, more formally, for all sets of sentences Γ and sentences φ, it is *not* the case that $\Gamma \therefore^{\text{empty}} A$. Even more formally, $C^{\text{empty}} = \emptyset$.

According to the empty relation, no argument is valid. Even the argument '$1 = 1$ therefore $1 = 1$' is invalid, since it is not the case that $\{1 = 1\} \therefore^{\text{empty}} 1 = 1$.

- The **trivial relation** is a consequence relation. Define C^{trivial} to be the relation between sets of sentences and sentences so that *all sentences are related to all sets of sentences.* Or, more formally, for all sets of sentences Γ and sentences φ, it *is* the case that $\Gamma \therefore^{\text{trivial}} A$.

According to the trivial relation, all arguments are valid. Even the argument '$1 = 1$ therefore $1 \neq 1$' is valid, since it is the case that $\{1 = 1\} \therefore^{\text{trivial}} 1 \neq 1$.

- The **membership relation** is a consequence relation. Define C^{member} to be the relation between sets of sentences and sentences so that $\Gamma \therefore^{\text{member}} A$ if and only if A is a member of Γ. Or, formally $C^{\text{member}} = \{\langle \Gamma, \varphi \rangle \mid \varphi \in \Gamma\}$.

According to the membership relation, only very boring arguments are valid: those where the conclusion is already a premiss. If your logic is the membership relation, you are never going to learn anything new by logical inquiry.

We can now consider, in the abstract, what kind of properties consequence relations can have, which of these properties we find desirable or useful, and which consequence relations have these properties. Here are some interesting ones.

Definition 7. A consequence relation C is:

- **trivial** iff$_{\text{Def}}$ it is the same as C^{trivial}. (A great many interesting-looking relations turn out to be this.)
- **monotone** iff$_{\text{Def}}$ if $\Gamma \therefore^{C} \varphi$, then for every superset Γ' of Γ, it is the case that $\Gamma' \therefore^{C} \varphi$.
- **compact** iff$_{\text{Def}}$ $\Gamma \therefore^{C} \varphi$ if and only if there is a finite subset Γ' of Γ with $\Gamma' \therefore^{C} \varphi$.

We are particularly interested in the properties of two particular consequence relations. You may have heard either or both of the following 'definitions' of valid argument.

- An argument is valid iff$_{\text{Def}}$ in all situations where the premisses are true, the conclusion is true as well. (The argument necessarily preserves truth.)
- An argument is valid iff$_{\text{Def}}$ it proceeds by steps and every step instantiates a (primitively valid) argument form. (There is a proof of the conclusion from the premisses.)

The first is a rough definition the **semantic** theory of consequence, as it appeals to the *truth-conditions* of the parts of the arguments. The second is a broad outline of the **syntactic** theory of consequence, as it appeals to the *forms* of the parts of the argument. Instead of C^{semantic} we will write $\models$ and instead of $C^{\text{syntactic}}$ we will write $\vdash$. But we will get to this.

In this course, we do two things. First we will develop **precise definitions** for both notions of valid argument for propositional logic and predicate logic. Then, we will demonstrate the following:

- Both semantic and syntactic consequence are compact. This is very hard to prove for semantic consequence, but very easy for syntactic consequence. Can you see why?
- Our precise version of syntactic consequence is **sound** with respect to our precise version of semantic consequence. That is, when we proceed by valid argument forms, we are also preserving truth.
- Our precise version of syntactic consequence is **complete** with respect to our precise version of semantic consequence. That is, all truth-preserving arguments can be done step by step, syntactically.

Take a moment to appreciate how surprising completeness is. On the face of it, it seems extremely difficult (perhaps impossible) to ever verify that in *all* (metaphysically possible) situations something is the case.

However, checking whether a syntactic step-by-step argument is valid is extremely easy: we just need to look at all of the steps and check whether they actually instantiate one of a small set of primitive forms. So with completeness, we have a very easy way (checking an argument) to do something very difficult (checking all situations).

But how will we even get started *proving* things about consequence relations?

1.3 Recursive definitions and inductive proofs

Our first problem is that we haven't yet said anything about what these 'sentences' are in the definition of consequence relation. Natural language sentences are no good for our purposes, since they can be ambiguous. We do not want to deal with this and hence are looking to define a formal language in which we can state (unambiguous) sentences for which we can then proceed to define consequence relations. It is important that we are *extremely precise* when stating these definitions.

We have, however, a problem: most likely, our formal language will allow us to define infinitely many sentences. (Think about it for yourselves whether our natural language supports infinitely many sentences as well.) So when we want to state the set of sentences we are interested in, we cannot simply list them all.

Let's start with a simpler version of the same problem. There are infinitely many natural numbers. How can we define the *set of all* natural numbers? If you ask someone what the natural numbers are, they might respond with something like the following.

> *you know, 0, 1, 2, 3 and so on.*

or if they had read the previous section

> $\{0, 1, 2, 3, ...\}$

But neither 'and so on' nor '...' are sufficiently *precise*. What if I tell you that this *actually* defines the set of numbers that are the sum of three squares? You know,

> $0, 1, 2, 3, 4, 5, 6, 8, 9, 10, 11, 12, 13, 14, 16, 17$ *and so on.*

What right would you have to tell me that I am wrong?

Towards a solution, it is useful to consider *why* the natural numbers are infinite. The reason is that every number has a successor. Or, if we want to be very precise:

Theorem 1.1. *There are infinitely many natural numbers.*

Proof. Assume towards a *reductio* that there are only finitely many natural numbers. Let N be their maximum, i.e. the biggest natural number. Let $N + 1$ be the successor of N. Note that $N + 1 > N$. But N was assumed to be the biggest natural number, so $N \geq N + 1$. This is a contradiction. By *reductio* there aren't only finitely many natural numbers, so there are infinitely many. □

Quite a few tacit assumptions go into this argument, but let's not worry about this for now. The crucial observation is that the fact that *every natural number has a successor.* This is the root of our problems with infinity here. Now, it is not difficult to see that something like a converse of this is also true: *every natural number except 0 is a successor of another natural number.* So the following is a good idea to define the natural numbers:

> The natural numbers are exactly 0 and all numbers obtained by iteratively taking successors.

'Iteratively' is already *much* better than 'and so on' but there may still be some confusion about what *precisely* is meant by this. Such confusions can be put to rest as follows:

- (i) 0 is a natural number.
- (ii) for all n: if n is a natural number, then the successor of n is a natural number.
- (iii) nothing else is a natural number. (More precisely: there is no set s with $0 \in s$ and (if $n \in s$, then $n + 1 \in s$) that is a proper subset of the natural numbers.)

(i), (ii) and (iii) together are a **recursive definition** of the natural numbers. We will denote the set of all natural numbers with $\mathbb{N}$.

Recursive definitions are a very powerful method for defining (possibly infinite) sets of things. They always consist of an *initial step* like (i) that explicitly states some members of the set and a *recursive step* like (ii) that states how to find further members by applying some operation to things that are already members. The condition (iii) then ensures that we get the set *only* containing the initial elements and everything obtainable from them by recursion.

This is how you make it precise what you mean by 'and so on' in definitions. But recursive definitions have another advantage: they allow you to prove things about their members using the (extremely powerful) technique of *induction arguments*. Induction arguments are the formal versions of 'and so on' in proofs.

Suppose you want to show the following.

Theorem 1.2. *For all natural numbers n: the sum of n and all natural numbers smaller n is $\frac{n\cdot(n+1)}{2}$.*

Perhaps you start to go about proving this as follows.

- 0: the sum of 0 and all natural numbers smaller 0 is just 0, which is indeed also $\frac{0\cdot 1}{2}$.
- 1: the sum $0 + 1 = 1$, which is indeed $\frac{1\cdot 2}{2}$.
- 2: the sum $0 + 1 + 2 = 3$, which is indeed $\frac{2\cdot 3}{2}$.
- 3: the sum $0 + 1 + 2 + 3 = 6$, which is indeed $\frac{3\cdot 4}{2} = \frac{12}{2} = 6$.
- *and so on*

This isn't a very convincing 'and so on'. We clearly have to do better. And indeed we can, by using the method of *proof by induction.*

Proof by Induction (also called Mathematical Induction)

To prove that all natural numbers have property P, it suffices to show the following:

- **Base case:** Show that $P(0)$.
- **Inductive step:** Assume that for a fixed but arbitrary n it is the case that $P(n)$ (this is the **Induction Hypothesis**). Using this assumption, show that $P(n+1)$.

You don't have to believe me that this proof method works. We can *prove that it works.*

Theorem 1.3. *Let P be a property. Suppose we have shown the Base case and Inductive step in a proof by induction. Then for all natural numbers n, $P(n)$.*

Proof. Towards a *reductio* assume that not all natural numbers have property P. Let n be the smallest natural number that does not have P. By the definition of the natural numbers, we are in one of two cases.

- Case 1: $n = 0$. According to the Base case, 0 has property P. Contradiction to our assumption that n does not have property P.
- Case 2: there is a number m such that $n = m+1$. Because n is the smallest number without property P, it must be the case that m has property P. Because in the Inductive step, we have phrased the induction hypothesis for arbitrary natural numbers, we can use it to infer $P(m+1)$ from $P(m)$. But $P(m+1)$ just is $P(n)$. Contradiction to our assumption that n does not have property P.
- By condition (iii) there are no other cases.

Thus we reach a contradiction in all cases. Hence all natural numbers have property P. □

Now we can prove Theorem 1.2.

Proof. Show by induction that for all natural numbers n, the sum of n and all natural numbers smaller n is $\frac{n\cdot(n+1)}{2}$.

- Base case. If the sum of 0 and all natural numbers smaller 0 is 0, which is also $\frac{0\cdot 1}{2}$.
- Inductive step. The induction hypothesis is: for a fixed but arbitrary n it is the case that the sum of n and all natural numbers smaller n is $\frac{n\cdot(n+1)}{2}$. It is to show that the sum of $n+1$ and all natural numbers smaller $n+1$ is $\frac{(n+1)\cdot(n+2)}{2}$.

  We know that the sum of $n+1$ and all natural numbers smaller $n+1$ can be written as follows: the sum of $n+1$ and n and all natural numbers smaller n. By the induction hypothesis, this can be written as $n+1+\frac{n\cdot(n+1)}{2}$.

  Now some arithmetic: $n+1+\frac{n\cdot(n+1)}{2} = \frac{2(n+1)}{2} + \frac{n\cdot(n+1)}{2} = \frac{2(n+1)+n\cdot(n+1)}{2} = \frac{(n+1)(2+n)}{2}$.

By induction, we are done. □

When you write an induction proof, it is very important to **cover the base case** (it is not always easy) and to **state the induction hypothesis** (it is not always obvious what it is).

2 Propositional Logic

Propositional logic abstracts away from the *contents* of sentences and deals only in how the *sentential connectives* (for example: ‘and’, ‘or’) contribute to logical consequence.

2.1 The language of propositional logic

As said, our first goal is to define a **precise language** for propositional logic. Again we need some basic notions first.

Definition 8 (Alphabet and Word)**.** An **alphabet** is a set of symbols. A **word** over an alphabet is any ordered set containing only symbols from the alphabet.

When we want to be very precise, we write symbols in quotes so that we do not confuse the symbol ‘n’ with the variable n.

For example, if our alphabet is $\{\text{‘}a\text{’}, \text{‘}b\text{’}, \text{‘}c\text{’}\}$ then the following are words: $\langle \text{‘}b\text{’}, \text{‘}c\text{’} \rangle$, $\langle \text{‘}c\text{’} \rangle$, $\langle \text{‘}b\text{’}, \text{‘}b\text{’}, \text{‘}a\text{’} \rangle$ and $\langle \text{‘}a\text{’}, \text{‘}a\text{’}, \text{‘}a\text{’}, \text{‘}a\text{’}, \text{‘}a\text{’} \rangle$ (there’s infinitely many more). In almost all cases, it is clear that we are talking about symbols and their sequences, so we will make our lives a bit easier by being lazy about the proper notation for words over alphabets. We will write words as bc, c, bba, $aaaaa$ and so on.

Object language and meta-langugae

The reason for these quotes is that we are in danger of confusing the language *that we use to describe words and alphabets* (the language in which these notes are written) with the language *that we are formally defining as words over an alphabet* (the language that we are going to prove things about).

The language that we are formally defining and going to prove things about is the **object language** (as it is the object of our study). The language we are *using to* study the object language is the **meta-language**.

There is nothing mysterious about this. Object languages are mathematical objects like any other (they are not that different from numbers). It’s just that both contain letters and symbols and words and it is sometimes ambiguous which language we mean when using a letter or a symbol or a word.

Recall the operation for concatenation $^\frown$. With our lazy notation, $abc^\frown ccba = abcccba$.

Now we are ready to define the language of propositional logic. It consists of two building blocks.

Definition 9. The **alphabet of propositional logic** is the union of the following three sets.

- An infinite set $\mathtt{At}$ of *propositional atoms* (these abstract away from actual sentences). For each natural number n, let p_n be an atom and for convenience also let p, q and r be atoms. Moreover let the symbol $\bot$ be an atom (this is *falsum* and will denote falsity).

  Formally, $\mathtt{At} = \{p_n \mid n \in \mathbb{N}\} \cup \{p, q, r\} \cup \{\bot\}$.

- The set of *sentential connectives*: $\{\wedge, \vee, \rightarrow, \neg\}$.

  These correspond to the sentential connectives of natural language: ‘and’ (conjunction, $\wedge$), ‘or’ (disjunction, $\vee$), ‘if...then’ (conditional, $\rightarrow$) and ‘not’ (negation, $\neg$).

Negation is called an *unary connective* because it modifies a single sentence; the others are called *binary connectives* because they connect two sentences.

- The set of *parentheses*: $\{(,)\}$.

Many words over this alphabet are not what we would understand to be a formula. For example $(((($, $((pq)$, $\wedge\wedge\wedge r$ are words over the alphabet of propositional logic. We still need some sort of syntax to sort out the good words (the formulae) from the bad words.

Definition 10. Define by recursion the set `wff` of **well-formed formulae of propositional logic**.

i. For all atoms $a \in$ `At`, $\langle a\rangle \in$ `wff`.

ii. If $A \in$ `wff`, then also $\neg^\frown A \in$ `wff`.

iii. If $A \in$ `wff` and $B \in$ `wff`, then also $({}^\frown A^\frown \wedge^\frown B^\frown) \in$ `wff`, $({}^\frown A^\frown \vee^\frown B^\frown) \in$ `wff` and $({}^\frown A^\frown \rightarrow^\frown B^\frown) \in$ `wff`.

iv. Nothing else is a member of `wff`.

Note that written out non-lazily, the conjunction term in (iii) would be this:

if $A \in$ `wff`, then also $\langle \text{‘}(\text{’}\rangle^\frown A^\frown\langle \text{‘}\wedge\text{’}\rangle^\frown B^\frown\langle \text{‘})\text{’}\rangle \in$ `wff`.

Make sure you understand why A is not in quotes or angle brackets. The reason is that A is a *variable* (in the meta-language) *for* a word in the object language. Similarly, a in (i) is a variable for an atom.

We will now be even more lazy: instead of $\neg^\frown A$ just write $\neg A$, instead of $({}^\frown A^\frown \wedge^\frown B^\frown)$ just write $(A \wedge B)$ and the same for the other connectives.

Some examples on how to use this definition.

- $(p \wedge q)$ is a member of `wff`. Proof: p and q are atoms, so they are in `wff`. Hence by (iii), $(p \wedge q)$ is in `wff`.
- $p \wedge q$ is not a member of `wff`. Proof: by (iv), one must be able to construct $p \wedge q$ from (i), (ii) or (iii). Because this is a sequence with more than one member, it cannot be case (i). Because this sequence does not start with $\neg$, it cannot be (ii). Because this sequence does not start with $($ it cannot be (iii).

We can now start proving some substantial things about the well-formed formulae of propositional logic (for short: the wffs). Our goal is to prove the Unique Construction Theorem stating that the members of `wff` are structurally unambiguous.

Theorem 2.1 (Unique Construction). *Let $A \in$ `wff`. Then* exactly one *of the following is the case.*

1. *$A = \langle a\rangle$ for some $a \in$ `At`.*
2. *There is a unique $B \in$ `wff` such that $A = \neg B$.*
3. *There are unique $B \in$ `wff` and $C \in$ `wff` such that $A = (B \wedge C)$.*
4. *There are unique $B \in$ `wff` and $C \in$ `wff` such that $A = (B \vee C)$.*
5. *There are unique $B \in$ `wff` and $C \in$ `wff` such that $A = (B \rightarrow C)$.*

This may seem very simple, but it really is not. Not just *any* recursive definition with multiple recursive steps leads to unique constructions. For example, if we had not used parentheses at all in our definition of the well-formed formulae of propositional logic, $\neg p \wedge q$ would be well-formed, but it would not have a unique constructions: it could either be constructed from the formulae $\neg p$ and q using the recursive step for $\wedge$, or could be constructed from the formula $p \wedge q$ using the recursive step for $\neg$. We need unique constructions to later assign every well-formed formula a non-ambiguous truth condition.

We first need a little Lemma with a long-ish proof. Given two ordered sets s and t, say that s is a **proper initial segment** of t if there is a non-empty ordered set s' such that $s^\frown s' = t$.

Lemma 2.2. *No proper initial segment of a wff is a wff.*

Proof. Note that the wffs are ordered sets. We can associate each ordered set with its **length**: the number of elements.

Now, for *reductio* assume that there is a wff that has a proper initial segment that also is a wff. Let A be such a wff with *minimal length*. That is for all wffs B that are shorter than A, it is the case that no proper initial segment of B is a wff.

It is clear that A cannot have length 1: any proper initial segment must have a shorter length and hence have length 0. But there are no wffs of length 0. Thus A must be formed by (ii) or (iii).

- Case 1. $A = \neg B$ for some wff B. By assumption, there is an initial segment s of A that is a wff. Note that s must start with $\neg$. Now let s' be the ordered set such that $\neg^\frown s' = s$ (i.e. s without the first element). Then s' is a proper initial segment of B.

  But if s is a wff and $s = \neg s'$, then s' must be a wff. So s' is a proper initial segment of B and s' is a wff. But B is shorter than A, so this contradicts our assumption that A has minimal length.

- Case 2. $A = (B \wedge C)$. By assumption, there is an initial segment s of A that is a wff. Note that s must start with (. Thus, s must be formed by (iii), so there are wffs D and E such that $s = (D \wedge E)$ or $s = (D \vee E)$ or $s = (D \to E)$.

  Look at B and D. We can distinguish three cases.

  - Case 2.1: D is a proper initial segment of B. Then B is a shorter formula than A that has a proper initial segment that is a formula. This contradicts our assumption that A is minimal with that property.
  - Case 2.2: B is a proper initial segment of D. Then D is a shorter formula than A (because D is shorter than s is shorter than A) that has a proper initial segment that is a formula. This contradicts our assumption that A is minimal with that property.
  - Case 2.3: $D = B$. Then we know that $s = (B \wedge E)$. Because we also know that s is a proper initial segment of $(B \wedge C)$ it follows that $E)$ is a proper initial segment of $C)$. But this means that E is a proper initial segment of C. Thus C is a wff that has a proper initial segment that is also a wff. This contradicts our assumption that A is minimal.

In both cases, we reach a contradiction. Hence by *reductio*, there is no wff that has a proper initial segment that is also a wff. □

If this proof technique of choosing something minimal looks familiar from how we proved the efficacy of Proof by Induction, this is a good reason. Because we have defined the well-formed formulae recursively, we define an inductive proof technique.

Structural Induction (or: induction over the complexity of the formulae)

To prove that all well-formed formulae have property P, it suffices to show the following.

- **Base case:** Show that $P(A)$ is the case for all $A \in \mathtt{At}$.
- **Inductive steps:**
  - Negation: Assume that for a fixed but arbitrary $A \in \mathtt{wff}$ it is the case that $P(A)$ (this is the **Induction Hypothesis** for this step). Using this assumption, show that $P(\neg A)$.
  - Binary connectives: Assume that for fixed but arbitrary $A \in \mathtt{wff}$ and $B \in \mathtt{wff}$ it is the case that $P(A)$ and $P(B)$ (this is the **Induction Hypothesis** for this step). Using this assumption, show that $P((A \wedge B))$, $P((A \vee B))$ and $P((A \to B))$.

We can prove that Structural Induction works very similarly to how we have proved that Mathematical Induction works.

Theorem 2.3. *Let P be a property. Suppose we have shown the Base case and all Inductive steps in a Structural Induction. Then for all $A \in \mathtt{wff}$, $P(A)$.*

Proof. Towards a *reductio* assume that there is a wff without the property P. It follows from this assumption that there is a natural number n so that there is a wff of length n without property P and all wffs without property P are at least n symbols long. (Put differently: n is the minimum length of a counterexample to all wffs having P).

Let A be a wff with length n that does not have property P. We can distinguish the following cases.

i. A is an atom. But then A has property P by the Base case. Contradiction.

ii. $A = \neg B$ for some $B \in \mathtt{wff}$. Because the length of B is smaller than n, we know that $P(B)$. But from the Inductive step for negation, we can prove that $P(A)$ from $P(B)$. So $P(A)$. Contradiction.

iii. $A = (B \vee C)$ for some $B \in \mathtt{wff}$ and $C \in \mathtt{wff}$. Because the lengths of B and C are smaller than n, we know that $P(B)$ and $P(C)$. But from the Inductive step for disjunction, we can prove that $P(A)$ from $P(B)$ and $P(C)$. So $P(A)$. Contradiction.

iv. $A = (B \wedge C)$ for some $B \in \mathtt{wff}$ and $C \in \mathtt{wff}$. Because the lengths of B and C are smaller than n, we know that $P(B)$ and $P(C)$. But from the Inductive step for conjunction, we can prove that $P(A)$ from $P(B)$ and $P(C)$. So $P(A)$. Contradiction.

v. $A = (B \to C)$ for some $B \in \mathtt{wff}$ and $C \in \mathtt{wff}$. Because the lengths of B and C are smaller than n, we know that $P(B)$ and $P(C)$. But from the Inductive step for conditionals, we can prove that $P(A)$ from $P(B)$ and $P(C)$. So $P(A)$. Contradiction.

There are no more cases, so we reach a contradiction. By *reductio*, all wffs have property P. □

Structural Induction is a *very* powerful technique. We can use it to prove the Unique Construction Theorem.

Proof of Theorem 2.1. It is easy that every well-formed formula can be written as one of (1)–(5), but the difficult part is to show that *exactly one* of the five cases hold. We will show both at the same time this by Structural Induction.

- Base case: Suppose A is an atom. Then clearly, (1) is the case. None of (2)–(5) are the case, because in these cases, the first symbol in A would be a parenthesis or $\neg$. But A is an atom and hence is a sequence of one symbol that is not a parenthesis or $\neg$.
- Inductive steps:
  - Negation. Induction hypothesis: For a fixed but arbitrary $A \in \texttt{wff}$ it is the case that exactly one of (1)–(5) holds.

    From the IH, show that exactly one of (1)–(5) holds for $\neg A$.

    Clearly, (2) holds for $\neg A$. Moreover, (1) does not holds for $\neg A$ as $\neg A$ contains more than one symbol. Also, if either of (3), (4) or (5) would hold, $\neg A$ would start with a parenthesis. But it does not.

    Finally, we need to check that if $\neg A = \neg B$, then also $A = B$. But this is trivial.
  - Conjunction. Induction hypothesis: For fixed but arbitrary $A \in \texttt{wff}$ and $B \in \texttt{wff}$ it is the case that exactly one of (1)–(5) holds.

    From the IH, show that exactly one of (1)–(5) holds for $(A \wedge B)$.

    Clearly, (3) holds for $(A \wedge B)$. Moreover, (1) does not hold for $(A \wedge B)$ as $(A \wedge B)$ contains more than one symbol and (2) does not hold because $(A \wedge B)$ does not start with $\neg$. We also need to rule out (4) and (5):

    * If (4) holds, there are wffs C and D such that $(A \wedge B) = (C \vee D)$. Note that it cannot be the case that $A = C$ because the in the ordered set $(A \wedge B)$ the next symbol after $(A$ is $\wedge$, not $\vee$. Thus it must be the case that either $(A$ is a proper initial segment of $(C$ or $(C$ is a proper initial segment of $(A$. But neither can be the case, because this would mean that A is a proper initial segment of C, or C is one of A. But by the Lemma, proper initial segments of wffs are not wffs.
    * If (5) holds, there are wffs C and D such that $(A \wedge B) = (C \to D)$. This is exactly as the case for $\vee$.

    It remains to show that there are no $C \neq A$ and $D \neq B$ such that $(A \wedge B) = (C \wedge D)$. Again, this cannot be the case because if $C \neq A$ then either C is a proper initial segment of A or C is a proper initial segment of A, which is ruled out by the Lemma. Thus $C = A$. But then immediately also $D = B$.

  - The inductive steps for disjunction and conditionals are analogous to conjunction.

This concludes the induction. □

It is legitimate (also on the homeworks and in proofs you will write after this course) to claim that some inductive steps are analogous to others. But be **very careful when claiming analogy**, as sometimes things that look analogous may not be—and for very non-obvious reasons.

The upshot of the Unique Construction Theorem is that we can now define properties of wffs by appealing to how they are constructed. One useful notion is the main operator of a formula.

Definition 11 (Main Operator)**.** Let $A \in \texttt{wff}$. If A is not an atom, is **main operator** is defined as follows:

- $\neg$ if there is a $B \in \texttt{wff}$ with $A = \neg B$.
- $\wedge$ if there are $B \in \texttt{wff}$ and $C \in \texttt{wff}$ with $A = (B \wedge C)$.
- $\vee$ if there are $B \in \texttt{wff}$ and $C \in \texttt{wff}$ with $A = (B \vee C)$.
- $\rightarrow$ if there are $B \in \texttt{wff}$ and $C \in \texttt{wff}$ with $A = (B \rightarrow C)$.

Note that without Unique Construction, it could be ambiguous what "the" main operator of a formula is.

Even more usefully, Unique Construction allows us to *recurse on* the construction of the formulae. An example for such a recursive definition is the definition of the *subformulae* of a wff.

Definition 12 (Subformulae)**.** The function sf maps every wff to the **set of its subformulae**. It is defined as follows:

i. Base: $\mathrm{sf}(A) = \{A\}$ if A is atomic.
ii. Recursion step negation: $\mathrm{sf}(A) = \{A\} \cup \mathrm{sf}(B)$ if there is a $B \in \texttt{wff}$ with $A = \neg B$.
iii. Recursion for the binary connectives: $\mathrm{sf}(A) = \{A\} \cup \mathrm{sf}(B) \cup \mathrm{sf}(C)$ if there is a $B \in \texttt{wff}$ with $A = (B \wedge C)$ or $A = (B \vee C)$ or $A = (B \rightarrow C)$.

Again, it is due to Unique Construction that we know for every A which recursive step we are in. Here's an example of how this definition works:

- $\mathrm{sf}(((p \wedge q) \wedge r)) = \{((p \wedge q) \wedge r)\} \cup \mathrm{sf}((p \wedge q)) \cup \mathrm{sf}(r)$. To determine this, we need to compute $\mathrm{sf}((p \wedge q))$ and $\mathrm{sf}(r)$

  We know that $\mathrm{sf}((p \wedge q)) = \{(p \wedge q)\} \cup \mathrm{sf}(p) \cup \mathrm{sf}(q)$. To determine this, we need to compute $\mathrm{sf}(p)$ and $\mathrm{sf}(q)$. But we have reached base cases here, so $\mathrm{sf}(p) = \{p\}$ and $\mathrm{sf}(q) = \{q\}$. Putting everything together, $\mathrm{sf}((p \wedge q)) = \{(p \wedge q, q, p)\}$.

  For $\mathrm{sf}\, r$ we already are in a base case; $\mathrm{sf}(r) = \{r\}$.

  Thus, $\mathrm{sf}(((p \wedge q) \wedge r)) = \{((p \wedge q) \wedge r), (p \wedge q), p, q, r\}$.

This may appear to be a lot of effort to compute something that you might be able to determine just by looking at a formula. The advantage of having gone through Unique Construction in order to be able to define a function like sf is as follows. In many proofs, we may be talking about a fixed but unknown formula

A. We may moreover want to talk about the set of subformulae of A. The recursive definition of sf allows us to do so without knowing which concrete formula the variable A stands for.

2.2 Truth tables and valuations

We are now ready to define the meaning of the connectives. As said, propositional logic abstracts away from the contents of concrete sentences. All we are interested in is what the connectives contribute when the sentences they connect are true or false.

Definition 13 (Truth Values)**.** The set $\{0, 1\}$ is the set of **truth values**. 1 stands for truth and 0 for falsity.

Definition 14 (Valuation)**.** A **valuation** is a function $\mathcal{V}$ that maps every atom to a truth value. In short, $\mathcal{V} : \texttt{At} \to \{0, 1\}$. The value of Falsum is always falsity, i.e. $\mathcal{V}(\bot) = 0$.

Valuations assign truth values to the atoms. The meaning of the connectives is given by how such assignments are **extended** to complex formulae.

Definition 15 (Extensions of Valuations)**.** Let $\mathcal{V}$ be a valuation. The **extension** of $\mathcal{V}$ is the function from wffs to truth values (short: $\mathcal{V}^* : \texttt{wff} \to \{0, 1\}$) defined by the following recursion.

i. Base: if $A \in \texttt{wff}$ is an atom, then $\mathcal{V}^*(A) = \mathcal{V}(A)$.

   (Note a slight abuse of notation here: technically, an atomic formula A is an ordered set containing a single atom, but not itself an atom. So we should *really* write $\mathcal{V}^*(A) = \mathcal{V}(a)$ where $A = \langle a \rangle$ for some $a \in \texttt{At}$. But it is harmless to ignore this.)

ii. The recursive steps are given by the following **truth tables**:

Negation:

$\mathcal{V}^*(\neg B)$	$\mathcal{V}^*(B)$
0	1
1	0

Conjunction:

$\mathcal{V}^*((B \wedge C))$	$\mathcal{V}^*(B)$	$\mathcal{V}^*(C)$
1	1	1
0	1	0
0	0	1
0	0	0

Disjunction:

$\mathcal{V}^*((B \vee C))$	$\mathcal{V}^*(B)$	$\mathcal{V}^*(C)$
1	1	1
1	1	0
1	0	1
0	0	0

Conditional:

$\mathcal{V}^*((B \to C))$	$\mathcal{V}^*(B)$	$\mathcal{V}^*(C)$
1	1	1
0	1	0
1	0	1
1	0	0

Given a valuation $\mathcal{V}$ and a formula $A \in \texttt{wff}$, $\mathcal{V}^*(A)$ is the **truth value of A given** $\mathcal{V}$.

Due to our prior work, we know that for every valuation $\mathcal{V}$ this defines exactly one extension $\mathcal{V}^*$.

Using these definitions, we can start classifying formulae into some familiar categories.

Definition 16. A formula $A \in \texttt{wff}$ is

- a **tautology** if for all valuations $\mathcal{V}$, $\mathcal{V}^*(A) = 1$;

- **satisfiable** if there is a valuation $\mathcal{V}$ such that $\mathcal{V}^*(A) = 1$;
- a **contradiction** if it is not satisfiable.
- **contingent** if it is neither a tautology nor a contradiction.

But again we are facing a problem with infinity. Clearly, there are infinitely many valuations $\mathcal{V}$. So if we want to show that some formula A is a tautology, it seems that we have to check all of them.

There is an intuitive solution to this problem. When we are looking for the truth value of A given $\mathcal{V}$, it only really matters what $\mathcal{V}$ is assigning to the propositional atoms that occur in A. That $\mathcal{V}$ is also assigning a truth value to the infinity of other atoms is just useless extra information. Note that only finitely many atoms occur in any given formula A and there are only finitely many ways to assign a truth value to all of them. As usual, this kind of reasoning can be made very, very precise.

First, we define the set of atoms occurring in a formula. We do so recursively.

Definition 17 (Atoms in a Formula)**.** The function at maps every wff A to the **set of atoms occurring in** A. It is defined as follows:

i. Base: $\text{at}(A) = \{A\}$ if A is itself atomic.
(Again the fact that technically A is an ordered set containing an atom, but not itself an atom, is ignored here and henceforth.)
ii. Recursion step negation: $\text{at}(A) = \text{at}(B)$ if there is a $B \in \texttt{wff}$ with $A = \neg B$.
iii. Recursion for the binary connectives: $\text{at}(A) = \text{at}(B) \cup \text{at}(C)$ if there are $B \in \texttt{wff}$ and $C \in \texttt{wff}$ with $A = (B \wedge C)$ or $A = (B \vee C)$ or $A = (B \rightarrow C)$.

Using this definition, we can prove the **Coinciding Valuations Theorem**.

Theorem 2.4 (Coinciding Valuations)**.** *Let A be a wff and let $\mathcal{V}_1$ and $\mathcal{V}_2$ be valuations such that for all $p \in \text{at}(A)$, $\mathcal{V}_1(p) = \mathcal{V}_2(p)$. Then $\mathcal{V}_1^*(A) = \mathcal{V}_2^*(A)$.*

Proof. By Structural Induction on A. Fix two valuations $\mathcal{V}_1$ and $\mathcal{V}_2$.

- Base: If A is an atom and for all $p \in \text{at}(A)$, $\mathcal{V}_1(p) = \mathcal{V}_2(p)$, then $\mathcal{V}_1^*(A) = \mathcal{V}_1(A)$ by definition, which is $\mathcal{V}_2(A)$ by assumption, which is $\mathcal{V}_2^*(A)$ by definition.
- Inductive step for negation. Induction hypothesis: For a fixed but arbitrary $A \in \texttt{wff}$ it is the case that if for all $p \in \text{at}(A)$, $\mathcal{V}_1(p) = \mathcal{V}_2(p)$, then also $\mathcal{V}_1^*(A) = \mathcal{V}_2^*(A)$.

  It is to show that if for all $p \in \text{at}(\neg A)$, $\mathcal{V}_1(p) = \mathcal{V}_2(p)$, then also $\mathcal{V}_1^*(\neg A) = \mathcal{V}_2^*(\neg A)$. We can distinguish two cases:

  - Case 1: it is not the case that for all $p \in \text{at}(\neg A)$, $\mathcal{V}_1(p) = \mathcal{V}_2(p)$. Then what is to show is vacuously true.
  - Case 2: it is the case that for all $p \in \text{at}(\neg A)$, $\mathcal{V}_1(p) = \mathcal{V}_2(p)$. Because we know that $\text{at}(\neg A) = \text{at}(A)$, it follows from the IH that $\mathcal{V}_1^*(A) = \mathcal{V}_2^*(A)$. By definition the values of $\mathcal{V}_1^*(\neg A)$ and

$\mathcal{V}_2^*(\neg A)$ depend *only* on the values of, respectively, $\mathcal{V}_1^*(A)$ and $\mathcal{V}_2^*(A)$. Because the latter are the same, the former are hence the same as well.

There are no more cases, and in both we have shown what was to show.

- Inductive step for conjunction. Induction hypothesis: For fixed but arbitrary $A \in \texttt{wff}$ and $B \in \texttt{wff}$ it is the case that (a) if for all $p \in \mathrm{at}(A)$, $\mathcal{V}_1(p) = \mathcal{V}_2(p)$, then also $\mathcal{V}_1^*(A) = \mathcal{V}_2^*(A)$; and (b) if for all $p \in \mathrm{at}(B)$, $\mathcal{V}_1(p) = \mathcal{V}_2(p)$, then also $\mathcal{V}_1^*(B) = \mathcal{V}_2^*(B)$.

  It is to show that if for all $p \in \mathrm{at}((A \wedge B))$, $\mathcal{V}_1(p) = \mathcal{V}_2(p)$, then also $\mathcal{V}_1^*((A \wedge B)) = \mathcal{V}_2^*((A \wedge B))$. We can distinguish two cases:

  - Case 1: it is not the case that for all $p \in \mathrm{at}((A \wedge B))$, $\mathcal{V}_1(p) = \mathcal{V}_2(p)$. Then what is to show is vacuously true.
  - Case 2: it is the case that for all $p \in \mathrm{at}((A \wedge B))$, $\mathcal{V}_1(p) = \mathcal{V}_2(p)$. Because we know that $\mathrm{at}((A \wedge B)) = \mathrm{at}(A) \cup \mathrm{at}(B)$, it follows that also for all $p \in \mathrm{at}(A)$, $\mathcal{V}_1(p) = \mathcal{V}_2(p)$ and for all $p \in \mathrm{at}(B)$, $\mathcal{V}_1(p) = \mathcal{V}_2(p)$.

    Thus, it follows from the IH that $\mathcal{V}_1^*(A) = \mathcal{V}_2^*(A)$ and $\mathcal{V}_1^*(B) = \mathcal{V}_2^*(B)$. As above, the values of $\mathcal{V}_1^*((A \wedge B))$ and $\mathcal{V}_2^*((A \wedge B))$ depend *only* on the values of, respectively, $\mathcal{V}_1^*(A)$ & $\mathcal{V}_1^*(B)$ and $\mathcal{V}_2^*(A)$ & $\mathcal{V}_2^*(B)$. Because the latter are pairwise the same, the former are hence the same as well.

  There are no more cases, and in both we have shown what was to show.

- The other inductive steps are analogous.

This concludes the induction. □

Using the Coinciding Valuations Theorem, we can determine whether a formula is tautological (etc.) in finite time. Given a formula A, first determine the set $\mathrm{at}(A)$. This set contains a finite number n of propositional atoms, so there are only 2^n possible assignments of truth values $V : \mathrm{at}(A) \to \{0, 1\}$. To know whether A is a tautology it now suffices to pick for each of these V a valuation $\mathcal{V}$ that agrees with V on the values of all atoms in $\mathrm{at}(A)$ and check $\mathcal{V}^*(A)$. (And similar for satisfiability, contradictoriness, contingency.) We can systematise this process in the Truth Table Method.

Truth Table Method

Given some A, find the set of its subformulae $\mathrm{sf}(A)$ and atoms $\mathrm{at}(A)$. Assign to every subformula a column in a table and to every possible function $V : \mathrm{at}(A) \to \{0, 1\}$ a row in this table.

Fill the columns for atomic formulae with the values assigned to them by V in every row.

Then determine (according to the truth tables) in each row the truth values of the subformulae of A that are formed directly from atoms. Continue determining the truth value of more complex subformulae until A is reached.

Here's an example. Above, we determined the subformulae of $((p \wedge q) \wedge r)$ to be $\mathrm{sf}(((p \wedge q) \wedge r)) = \{((p \wedge q) \wedge r), (p \wedge q), p, q, r\}$. The atoms are $\{p, q, r\}$. So we construct a table looking like this to conclude that $((p \wedge q) \wedge r)$ is satisfiable and contingent.

p	q	r	$(p \wedge q)$	$((p \wedge q) \wedge r)$		p	q	r	$(p \wedge q)$	$((p \wedge q) \wedge r)$		p	q	r	$(p \wedge q)$	$((p \wedge q) \wedge r)$
1	1	1				1	1	1	1			1	1	1	1	1
1	1	0				1	1	0	1			1	1	0	1	0
1	0	1				1	0	1	0			1	0	1	0	0
1	0	0			$\rightsquigarrow$	1	0	0	0		$\rightsquigarrow$	1	0	0	0	0
0	1	1				0	1	1	0			0	1	1	0	0
0	1	0				0	1	0	0			0	1	0	0	0
0	0	1				0	0	1	0			0	0	1	0	0
0	0	0				0	0	0	0			0	0	0	0	0

Here are some useful tautologies.

Theorem 2.5. *Let A be any wff. Then the following are tautologies.*

- Law of Identity*:* $(A \to A)$*.*
- Law of Excluded Middle*:* $(A \vee \neg A)$*.*
- Law of Non-Contradiction*:* $\neg(A \wedge \neg A)$*.*

These are the three traditional laws of logic.

2.3 Equivalence and Substitution

Some different formulas are *equivalent* in that they have the same truth conditions. Formally:

Definition 18 (Equivalence)**.** Well-formed formulae A and B are **equivalent** (write: $A \approx B$) iff$_{\text{Def}}$ for all valuations $\mathcal{V}$ it is the case that $\mathcal{V}^*(A) = \mathcal{V}^*(B)$.

Using the Truth Table Method we can check whether two formulae are equivalent. Here are some useful equivalences.

Theorem 2.6. *Let A, B and C be wffs.*

- Conjunction is associative: $((A \wedge B) \wedge C) \approx (A \wedge (B \wedge C))$*.*
- Conjunction is commutative: $(A \wedge B) \approx (B \wedge A)$*.*
- Disjunction is associative: $((A \vee B) \vee C) \approx (A \vee (B \vee C))$*.*
- Disjunction is commutative: $(A \vee B) \approx (B \vee A)$*.*
- Distributivity: $(A \wedge (B \vee C)) \approx ((A \wedge B) \vee (A \wedge C))$ *and*
  $(A \vee (B \wedge C)) \approx ((A \vee B) \wedge (A \vee C))$
- De Morgan's Laws*:* $\neg(A \wedge B) \approx (\neg A \vee \neg B)$ *and*
  $\neg(A \vee B) \approx (\neg A \wedge \neg B)$*.*
- $(A \vee B) \approx \neg(\neg A \wedge \neg B)$*.*
- $(A \wedge B) \approx \neg(\neg A \vee \neg B)$*.*

The proofs are all straightforward applications of the Truth Table Method and are left as exercises. **We will henceforth drop parentheses if their only purpose is to disambiguate equivalent formulae.**

For example, instead of $((p \wedge q) \wedge r)$ we will write $(p \wedge q \wedge r)$. We will also sometimes drop outermost parentheses, as they do not help to disambiguate anything, so we may also write $p \wedge q \wedge r$.

The important property of equivalent formulas is that they can be substituted for one another without affecting truth value.

Definition 19 (Substitution)**.** Let A and B be wffs and a be an atom. Define by recursion on A the **substitution of** B **for** a **in** A, written as $A[B/a]$.

- Base: if A is an atom, then $A[B/a] = B$ if $A = a$ and $A[B/a] = A$ otherwise.
- Recursive steps: if $A = \neg C$, then $A[B/a] = \neg^\frown C[B/a]$;
  if $A = C \wedge D$, then $A[B/a] = (C[B/a]^\frown \wedge^\frown D[B/a])$;
  if $A = C \vee D$, then $A[B/a] = (C[B/a]^\frown \vee^\frown D[B/a])$;
  if $A = C \rightarrow D$, then $A[B/a] = (C[B/a]^\frown \rightarrow^\frown D[B/a])$;

Using this definition, we can prove the following.

Theorem 2.7 (Substitution Theorem)**.** *Let A, B and C be well-formed formulae and a be an atom. If $A \approx B$, then $C[A/a] \approx C[B/a]$.*

Proof. Fix arbitrary A, B and a with $A \approx B$ and prove the theorem by induction on the construction of C.

- Base case: if C is an atom and $C \neq a$, then $C[A/a] = C = C[B/a]$ and $C \approx C$ trivially. If $C = a$, then $C[A/a] = A$ and $C[B/a] = B$ and $A \approx B$ by assumption.
- Inductive step negation. Suppose that $C = \neg D$. The induction hypothesis is: $D[A/a] \approx D[B/a]$. It is to show that $(\neg D)[A/a] \approx (\neg D)[B/a]$. By definition of substitution, this means that it is to show that $\neg^\frown D[A/a] \approx \neg^\frown D[B/a]$

  By the induction hypothesis $D[A/a]$ and $D[B/a]$ have the same truth value under all valuations $\mathcal{V}$. But then also their negations have the same truth values.
- Inductive step conjunction. Suppose that $C = D \wedge E$. The induction hypothesis is: $D[A/a] \approx D[B/a]$ and $E[A/a] \approx E[B/a]$. It is to show that $(D \wedge E)[A/a] \equiv (D \wedge E)[B/a]$. By definition of substitution, this means it is to show that $(D[A/a] \wedge E[A/a]) \approx (D[B/a] \wedge E[B/a])$.

  By the induction hypothesis $D[A/a]$ and $D[B/a]$ have the same truth value under all valuations $\mathcal{V}$ and $E[A/a]$ and $E[B/a]$ have the same truth value under all valuations. Then also $D[A/a] \wedge E[A/a]$ and $D[B/a] \wedge E[B/a]$ have the same truth value under all valuations.
- The other inductive steps are analogous.

This concludes the induction. □

This may not yet be *quite* what one expects from the slogan that *equivalent formulae are intersubstitutable.* We may define the substitution of one formula for another as follows. To define $C[B/A]$, find a formula C' and an atom $a \in \mathrm{at}(C')$ such that $C = C'[A/a]$. If there is such a C', let $C[B/A] =_{\text{Def}} C'[B/a]$;

else let $C[B/A] =_{\text{Def}} C$. It follows immediately from the Substitution Theorem that if $A \approx B$, then $C'[B/a] \approx C'[A/a]$ and therefore $C \approx C[B/A]$.

The following result will allow us to express equivalence in the object language.

Theorem 2.8. *Let A and B be wffs. $A \approx B$ if and only if $((A \to B) \wedge (B \to A))$ is a tautology.*

Proof. Let A and B be wffs.

$\Rightarrow$ left-to-right: Assume that $A \approx B$ and show that for every valuation $\mathcal{V}$, it is the case that $\mathcal{V}(((A \to B) \wedge (B \to A))) = 1$. So let $\mathcal{V}$ be any valuation. The value of $((A \to B) \wedge (B \to A))$ can be computed by the following table.

$\mathcal{V}^*(A)$	$\mathcal{V}^*(B)$	$\mathcal{V}^*(A \to B)$	$\mathcal{V}^*(A \to B)$	$\mathcal{V}^*(((A \to B) \wedge (B \to A)))$
1	1	1	1	1
1	0	0	1	0
0	0	1	1	1
0	1	1	0	0

But by assumption, $\mathcal{V}^*(A) = \mathcal{V}^*(B)$, so the second and fourth row are ruled out.

Thus $\mathcal{V}^*(((A \to B) \wedge (B \to A))) = 1$. As $\mathcal{V}$ was arbitrary, we conclude that for all $\mathcal{V}$, $\mathcal{V}^*(((A \to B) \wedge (B \to A))) = 1$.

$\Leftarrow$ right-to-left: Assume that $((A \to B) \wedge (B \to A))$ is a tautology. Show that for every valuation $\mathcal{V}$, $\mathcal{V}^*(A) = \mathcal{V}^*(B)$. So let $\mathcal{V}$ be arbitrary. We can apply the truth table method in a backwards way.

Look at the truth table in the previous step. By assumption, $\mathcal{V}^*(((A \to B) \wedge (B \to A))) = 1$, so we must be in the first or third rows. But in either case $\mathcal{V}^*(A) = \mathcal{V}^*(B)$. As $\mathcal{V}$ was arbitrary, this goes for all valuations $\mathcal{V}$. □

We can shorten the statement of this last result by introducing an *abbreviation*.

Definition 20. Let $(A \leftrightarrow B)$ be an abbreviation of $((A \to B) \wedge (B \to A))$.

That is, whenever we write $\leftrightarrow$, we keep in mind that we actually mean $((A \to B) \wedge (B \to A))$, but just don't want to write out this long formula. Another useful abbreviation is to write the *Verum symbol* $\top$ as an abbreviation of $\bot \to \bot$. Note that $\mathcal{V}^*(\top) = 1$ for all valuations.

2.4 Expressive Power

We have seen above that some connectives can be defined from others. This means that, for example, we would not strictly speaking *need* the symbol '$\vee$' in our language as we could also regard $A \vee B$ as an abbreviation of $\neg(\neg A \wedge \neg B)$.

This raises the question of how many connectives we actually need. Do we need *more* than the ones we have to express some interesting formulae? Note that there are $2^{(2^2)} = 16$ possible truth tables for binary connectives but we only have been using three. We could also imagine *trinary* connectives (like *if ... then ... else*). There are $2^{(2^3)} = 256$ possible trinary connectives. Do we need some of them?

We can make this question precise by using the following definition.

Definition 21 (Boolean Function)**.** Let P be a finite set of atoms. A **Boolean function on** P is a function f that maps every function from P to truth values to a truth value. Formally, $f : \{V \mid V : P \to \{0, 1\}\} \to \{0, 1\}$.

One may think of the Boolean functions as the possible truth tables. A Boolean function specifies how to obtain a truth value from any possible valuation of a finite set of input values. That's the kind of thing you can display in a truth table.

Given a formula A, we can define a corresponding Boolean function f_A on $\mathrm{at}(A)$. To wit: For all functions $V : \mathrm{at}(A) \to \{0, 1\}$ let $\mathcal{V}$ be any valuation that assigns something to all the atoms not in $\mathrm{at}(A)$ and define $f_A(V) = \mathcal{V}^*(A)$. Note that by the Coinciding Valuations Theorem, it does not matter what we assign to the atoms not occurring in A. So the value definition of $f_A(V)$ is unambiguously defined. We call f_A the Boolean function **expressed by** A.

So here is a natural desideratum: our language should be expressive enough so that we can express every Boolean function, i.e. so that every possible truth table can be written as an object-language formula. That is the maximum expressivity we can get out of propositional logic.

Definition 22 (Propositional Basis)**.** A subset L of $\{\bot, \neg, \wedge, \vee, \to\}$ is called a **propositional basis** iff$_{\text{Def}}$ when the language of propositional logic is defined only for symbols from L, then for every finite set of atoms P and every Boolean function f on P, there is a wff A with $\mathrm{at}(A) = P$ and $f = f_A$.

A propositional basis is also sometimes called a *functionally complete set*, but we would not want to confuse this use of 'complete' with the Completeness theorem that we show later.

Our first task is to show that our full language $\{\bot, \neg, \wedge, \vee, \to\}$ is a propositional basis. This follows from the slightly stronger result that every Boolean function is expressed by a formula in **normal form**.

The following definitions look a lot more scary than they actually are. First we want to be able to express formally that a formula can be written as a long conjunction or as a long disjunction.

- Given a number n and a set $\{A_i \mid i < n\}$ of formulae (sloppily: $\{A_0, ..., A_{n-1}\}$), we can define $\bigwedge_{i<n} A_i$, **the conjunction of** the A_i, by recursion on n.

  Base case $n = 1$: $\bigwedge_{i<1} A_i = A_0$. Recursive step: $\bigwedge_{i<n} A_i = (\bigwedge_{i<(n-1)} A_i) \wedge A_{n-1}$.

  Having the precise definition in our minds, we can be sloppy and write $\bigwedge_{i<n} A_i$ as $A_0 \wedge ... \wedge A_{n-1}$.
- Analogously, we can recursively define **the disjunction of** the A_i, $\bigvee_{i<n} A_i$ and allow ourselves to sloppily write it as $A_0 \vee ... \vee A_{n-1}$.

Now consider the following definition.

Definition 23 (Disjunctive Normal Form). A wff A is in **disjunctive normal form (DNF)** iff$_{\text{Def}}$ there is a number n and a set of formulae $\{A_0, ..., A_{n-1}\}$ such that:

i. $A = A_0 \vee ... \vee A_{n-1}$.
ii. for all A_i there is a number n_i and a set of formulae $\{B_{i,j} \mid j < n_i\}$ such that $A_i = B_{i,0} \wedge ... \wedge B_{i,n_i-1}$ is the conjunction of the B_{ij}; and
iii. for all i and j, $B_{i,j}$ is either an atom or the negation of an atom.

Using our formal definitions, we can write A concisely as $A = \bigvee_{i<n} \bigwedge_{j<n_i} B_{i,j}$.

That is, a formula A is in disjunctive normal form if it is a disjunction where every disjunct is a conjunction of atoms and negated atoms. Essentially, a formula in disjunctive normal form is just a list of all the conditions under which it is true: *(these atoms are true and those are false) or (these atoms are true and those are false) or ....*

Theorem 2.9. *For every finite set of atoms P and every Boolean function f on P, there is a formula A in disjunctive normal form with* $\mathrm{at}(A) = P$ *and* $f_A = f$.

Proof. Fix some arbitrary P and some Boolean function f on P. Let m be the number of atoms in P and write P as $P = \{p_j \mid j < m\}$.

Let T be the set of all $V : P \to \{0, 1\}$ such that $f(V) = 1$. Note that T is finite. So let n be the number of its members and write $T = \{V_i \mid i < n\}$.

Now, for every $i < n$ and $j < m$, we can define the formula $B_{i,j}$ as follows: Let $B_{i,j} = p_j$ iff$_{\text{Def}}$ $V_i(p_j) = 1$ and let $B_{i,j} = \neg p_j$ iff$_{\text{Def}}$ $V_i(p_j) = 0$.

For every $i < n$ define $A_i =_{\text{Def}} B_{i,0} \wedge B_{i,1} \wedge ... \wedge B_{i,m-1}$. Then finally define $A =_{\text{Def}} A_0 \vee ... \vee A_{n-1}$.

Clearly, A is in disjunctive normal form. We need to prove that $f = f_A$. So let V be any function $V : P \to \{0, 1\}$. It is to show that $f(V) = f_A(V)$.

- Case 1: $f(V) = 1$. In this case there is some $i < n$ such that $V = V_i$. Note that for any valuation $\mathcal{V}$ that agrees with V on all atoms in P it is the case that: for all $j < m$, $\mathcal{V}^*(B_{i,j}) = 1$ by the Truth Table for negation. Therefore, $\mathcal{V}^*(A_i) = 1$ by the Truth Table for conjunction. And hence, $\mathcal{V}^*(A) = 1$ by the Truth Table for disjunction. Thus $f_A(V) = 1$.
- Case 2: $f(V) = 0$. In this case there is *no* $i < n$ such that $V = V_i$. This means that V must disagree with every member of T on the truth value of at least one atom. Formally, this means that for every $i < n$ there is a $j < m$ such that $V(p_j) \neq V_i(p_j)$.

  Let $\mathcal{V}$ be any valuation that agrees with V on all atoms in P. By the above and the Truth Table for negation, for every i there is a j such that $\mathcal{V}^*(B_{i,j}) = 0$. Hence by the Truth Table for conjunction, for every i, $\mathcal{V}^*(A_i) = 0$ and hence by the Truth Table for disjunction, $\mathcal{V}^*(A) = 0$.

This concludes the proof. Note that if we were to be *very* formal here, we would use in both steps an induction over the definition of $\bigwedge$ and $\bigvee$. □

An important consequence of this theorem is that $\{\neg, \wedge, \vee\}$ is a propositional base because these symbols suffice to write disjunctive normal forms. Hence also $\{\bot, \neg, \wedge, \vee, \to\}$ is a propositional base, but Falsum and the conditional appear to be superfluous. Moreover, since we know that we can abbreviate $A \vee B$ by $\neg(\neg A \wedge \neg B)$ and abbreviate $A \wedge B$ by $\neg(\neg A \vee \neg B)$ we also know that $\{\neg, \wedge\}$ and $\{\neg, \vee\}$ are propositional bases.

This is our first **substantial result about logic**. We can express every possible truth table by only using two connectives. This is something that might not have been obvious when we began looking into the formal properties of logic. If you believe that the logic we use casually in our meta-language was adequately formalised in propositional logic you have also learned something about our actual logical practices: that in principle you could strike, say, 'or' from your vocabulary.

Another important upshot is that for every wff A there is a formula A^{DNF} in disjunctive normal form such that $A \approx A^{\text{DNF}}$. There is also a conjunctive normal form.

Definition 24 (Conjunctive Normal Form). A wff A is in **conjunctive normal form (CNF)** iff$_{\text{Def}}$ there is a number n and a set of formulae $\{A_0, \ldots, A_{n-1}\}$ such that:

i. $A = A_0 \wedge \ldots \wedge A_{n-1}$.
ii. for all A_i there is a number n_i and a set of formulae $\{B_{i,j} \mid j < n_i\}$ such that $A_i = B_{i,0} \vee \ldots \vee B_{i,n_i-1}$ is the disjunction of the B_{ij}; and
iii. for all i and j, $B_{i,j}$ is either an atom or the negation of an atom.

Using our formal definitions, we can write A concisely as $A = \bigwedge_{i<n} \bigvee_{j<n_i} B_{i,j}$.

The proof that every formula is equivalent to a formula in CNF is left as an exercise.

2.5 Semantic consequence

Now we can finally define the semantic consequence relation $\models$. Recall that we wanted this relation to express something like 'in all situations where all the premisses are true, also the conclusion is true.' Using valuations, we can express this now as follows.

Definition 25. Let Γ be a set of wff and A be a wff. $\Gamma \models A$ iff$_{\text{Def}}$ for all valuations $\mathcal{V}$, if $\mathcal{V}^*(B) = 1$ for all $B \in \Gamma$, then $\mathcal{V}^*(A) = 1$.

Or, in words, the premisses Γ semantically entail the conclusion A iff all valuations that make all premisses true also make the conclusion true. The valuations are here taking the place of the somewhat nebulous notion of a situation. As all we care about in propositional logic is whether a sentence is true or not, all we need to know about a situation is which sentences are true in it—which is provided by a valuation.

When Γ is a finite set that we (being somewhat sloppy) write as $\Gamma = \{A_0, A_1, \ldots, A_{n-1}\}$, we will usually write $A_0, \ldots, A_n \models A$ (leaving out the set brackets). When Γ is the empty set, we just write $\models A$ instead of $\emptyset \models A$.

Theorem 2.10. *Let A be a wff. Then $\models A$ iff A is a tautology.*

Proof. The left-hand-side says that for all valuations $\mathcal{V}$ so that $\mathcal{V}^*(B) = 1$ for all members B of $\emptyset$, it is the case that $\mathcal{V}^*(A) = 1$. But the empty set has no members, so trivially all valuations make all its members true. Thus what it says on the left-hand-side is just that all valuations $\mathcal{V}$ are such that $\mathcal{V}^*(A) = 1$. But that is just the definition of a tautology. □

We can push this result further to characterise consequences from finite sets of premisses by tautologies.

Theorem 2.11. *Let A and $A_0, \ldots, A_{n-1}$ be wffs. Then $A_0, \ldots, A_{n-1} \models A$ iff $(\bigwedge_{i<n} A_i) \to A$ is a tautology.*

Proof.

$\Rightarrow$ left-to-right. Prove the contrapositive: if $(\bigwedge_{i<n} A_i) \to A$ is not a tautology, then $A_0, \ldots, A_{n-1} \not\models A$. Assume that $(\bigwedge_{i<n} A_i) \to A$ is not a tautology, i.e. that for some valuation $\mathcal{V}$, $\mathcal{V}^*((\bigwedge_{i<n} A_i) \to A) = 0$.

Inspection of the Truth Table for → shows that therefore $\mathcal{V}^*(A) = 0$ and for all $i < n$, $\mathcal{V}^*(A_i) = 1$ (very formally, this is an induction on i). But this just means that there is a valuation that makes all the premisses $A_0, \ldots, A_{n-1}$ true, but the conclusion A false. So $A_0, \ldots, A_{n-1} \not\models A$.

$\Leftarrow$ right-to-left. Again prove the contrapositive: if $A_0, \ldots, A_{n-1} \not\models A$, then $(\bigwedge_{i<n} A_i) \to A$ is not a tautology. So assume that $A_0, \ldots, A_{n-1} \not\models A$, i.e. that for some valuation $\mathcal{V}$, for all $i < n$, $\mathcal{V}^*(A_i) = 1$ and $\mathcal{V}^*(A) = 0$.

Inspection of the Truth Table for $\wedge$ shows that then also $\mathcal{V}^*((\bigwedge_{i<n} A_i)) = 1$. But because $\mathcal{V} * (A) = 0$, it follows by the Truth Table for →, that $\mathcal{V}^*(\bigwedge_{i<n} A_i) \to A) = 0$. So $(\bigwedge_{i<n} A_i) \to A$ is not a tautology.

□

Thus, in principle, if we only ever have finite premisses, we don't strictly speaking need to define 'semantic consequence' and could instead just talk about certain conditionals being tautologies. We cannot, however, push the strategy to infinite sets of premisses. This is because our language contains only finite sentences, so there are no infinite conjunctions.

We will however later prove the compactness of $\models$. That is:

Theorem 2.12. *Let A be a wff and Γ be a set of wffs. Then $\Gamma \models A$ iff there is a finite subset $\Gamma' \subseteq \Gamma$ with $\Gamma' \models A$.*

From this it follows that $\Gamma \models A$ iff there is a finite subset $\{A_0, \ldots, A_{n-1}\} \subseteq \Gamma$ such that $(\bigwedge_{i<n} A_i) \to A$ is a tautology. This is the most general case of characterising semantic consequence by tautology. Note however that we cannot express 'there is a finite subset' in the object language.

So when it comes to infinite sets of premisses, the metalanguage is genuinely more powerful than the object language: there is no object language sentence that is a tautology if and only if $\Gamma \models A$. So there is something that we can 'say' in the object language (namely 'Γ semantically entails A') that we cannot say in the object language. This is a simple case of an observation that we will make time and again throughout this course: in certain, subtle ways, the meta-language is more expressive than the object language.

Moreover, there is a different notion we are already acquainted with that can help us give an alternative characterisation of what semantic consequence is: satisfiability. The notion of satisfiability straightforwardly extends to sets of formulae.

Definition 26. A set Γ of wffs is **satisfiable** if there is a valuation $\mathcal{V}$ such that for all $A \in \Gamma$, $\mathcal{V}^*(A) = 1$.

Then we can prove the following.

Theorem 2.13. *$\Gamma \models A$ iff $\Gamma \cup \{\neg A\}$ is not satisfiable.*

Proof.

$\Rightarrow$ left-to-right. We will prove the contrapositive: if $\Gamma \cup \{\neg A\}$ is satisfiable, then $\Gamma \not\models A$. So assume that $\Gamma \cup \{\neg A\}$ is satisfiable. Then there is a valuation $\mathcal{V}$ such that for all $B \in \Gamma$, $\mathcal{V}^*(B) = 1$ and $\mathcal{V}^*(\neg A) = 1$. The latter means that $\mathcal{V}^*(A) = 0$. So $\mathcal{V}$ is a valuation for which it is not the case that

$\Leftarrow$ right-to-left. We will prove the contrapositive: if $\Gamma \not\models A$, then $\Gamma \cup \{\neg A\}$ is satisfiable. So assume that $\Gamma \not\models A$. Then there is a valuation $\mathcal{V}$ such that for all $B \in \Gamma$, $\mathcal{V}^*(B) = 1$ and $\mathcal{V}^*(A) = 0$. The latter means that $\mathcal{V}^*(\neg A) = 1$. So $\mathcal{V}$ is a valuation for that shows that $\Gamma \cup \{\neg A\}$ is satisfiable. □

This vindicates our intuitive characterisation of $\models$ as 'it cannot be the case that the premisses are all true and the conclusion is false'.

2.6 The Natural Deduction Calculus

We now turn to finding a formal definition of syntactic consequence. Like the definition of semantic consequence tries to capture our pre-formal understanding of truth in situations, the definition of syntactic consequence should capture our pre-formal understanding of *proof* and *argument form*. We intend our formal results about consequence relations to reveal something about the practice of using logic. The task of defining syntactic consequence is hence best understood as a *modelling* task: we are looking for a formal model of what we are doing when we are proving things. We call such a model a *calculus*.

The *natural deduction calculus* was developed by Gerhard Gentzen with the specific ambition to come as close as possible to actual reasoning. It consists of *inference rules* that associate (possibly multiple) premisses with a single conclusion. These rules are intended to capture the *smallest discernible steps* that one can take in a proof, so that chaining them together will allow us to phrase the complex mathematical arguments we are becoming used to.

We will write inference rules in a two-dimensional notation in which the premisses are separated by a vertical line from the conclusion. For example, the inference rule of *modus ponens* has two premisses and can be written as follows.

$$\frac{A \to B \qquad A}{B}$$

This states that from $A \to B$ and A it is correct to infer B. Note that the intended reading of the *modus ponens* rule is *schematic*. The rule states that for *arbitrary* formulae A and B, one may infer B from $A \to B$

and A. That is, the following are all concrete instances of this rule.

$$\frac{p \to q \qquad p}{q} \qquad \frac{(p \vee q) \to \neg r \qquad (p \vee q)}{\neg r} \qquad \frac{\neg p \to r \qquad \neg p}{r}$$

Note that all these instances are obtained from the *modus ponens* scheme by replacing A and B with concrete formulae. The first example is obtained from the scheme by replacing A with p in *both* occurrences of A; and replacing B with q, also in both occurrences of B. It is not necessary to replace A and B with different formulae, so the following is also an instance of the *modus ponens* scheme.

$$\frac{p \to p \qquad p}{p}$$

The following however are *not* instances of *modus ponens*.

$$\frac{p \to q \qquad q}{q} \qquad \frac{p \to r \qquad p \wedge p}{r} \qquad \frac{\neg p \to r \qquad \neg p}{r \vee q}$$

The first one is not an instance of *modus ponens* because the second premiss does not match the antecedent of the conditional in the first premiss. Thus it is not formed by replacing A in the *modus ponens* scheme with a single concrete formula. It does not matter that you may have very good reason to think that q entails q. Whether or not something is an instance of *modus ponens* is only and exclusively about whether it has the right *form*.

The second one likewise is not an instance of *modus ponens* because the second premiss does not match the antecedent of the first premiss. Although we know that p and $p \wedge p$ are *equivalent* in our semantics, this plays no role here. Only form matters. Similarly, the third one is not an instance of *modus ponens* because its conclusion does not match the consequent of the conditional in the first premiss, so it is not formed by replacing B in the *modus ponens* scheme with a single concrete formula.

We are now looking for further inference rules that tell us how to reason with the connectives. Some of these rules are very simple. For example, the following rules *obviously* formalise appropriately how to reason with conjunction.

$$\frac{A \qquad B}{A \wedge B} \qquad \frac{A \wedge B}{A} \qquad \frac{A \wedge B}{B}$$

This means that if you have A and B (separately), you are entitled to infer their conjunction $A \wedge B$. Conversely, if have a conjunction $A \wedge B$, you are entitled to infer A and also entitled to infer B.

With these rules and *modus ponens* we can already write down some small *formal proofs*. We form proofs by chaining rules together, so that the premisses of one rule are the conclusions of another. For example, we can show that from the set of premisses $\{p_1 \wedge p_2, p_1 \to q, p_2 \to r\}$ it follows that $q \wedge r$. The proof looks as follows.

$$\dfrac{\dfrac{\dfrac{p_1 \wedge p_2}{p_1} \qquad p_1 \to q}{q} \qquad \dfrac{\dfrac{p_1 \wedge p_2}{p_2} \qquad p_2 \to r}{r}}{q \wedge r}$$

Note that we have used the premiss $p_1 \wedge p_2$ twice. This is permissible because we are always working with *sets* of premisses and there is no fact of the matter “how often” a formula occurs in the set of premisses. It only matters *that* a formula is a member of the set of premisses—and if it is it can be used in a proof at any point and arbitrarily often.

This way of displaying a formal argument is commonly called a *proof tree.* Proof trees can get very confusing when they grow larger, so it helps to indicate at every line which inference rule is instantiated in this step.

To do so, we will have to give our rules names. Which one we choose does not matter, but there are some helpful conventions.

Note that the rules for conjunction serve particular functions towards particular goals. If you want to reason *towards* a conjunction, the first rule tells you which premisses you need. The second and third rules for conjunction, however, tell you what you can get when you have a conjunction. Put differently, the first rule tells you how to *introduce* a conjunction into a proof; the second and third tell you how to *eliminate* a conjunction to get to its parts. So we will call the first the *introduction rule for conjunction* and the second and third the *elimination rules for conjunction.*

$$(\wedge\text{I.})\,\dfrac{A \qquad B}{A \wedge B} \qquad (\wedge\text{E.}_1)\,\dfrac{A \wedge B}{A} \qquad (\wedge\text{E.}_2)\,\dfrac{A \wedge B}{B}$$

Similarly, *modus ponens* can be thought of as the elimination rule for the conditional, as it also tells us how to get from a complex formula whose main operator is a conditional to its parts. So this is what we will call it.

$$(\to\text{E.})\,\dfrac{A \to B \qquad A}{B}$$

With these labels in place, we can make our proof tree a lot more readable.

$$\dfrac{\dfrac{\dfrac{p_1 \wedge p_2}{p_1}\,(\wedge\text{E.}_1) \qquad p_1 \to q}{q}\,(\to\text{E.}) \qquad \dfrac{\dfrac{p_1 \wedge p_2}{p_2}\,(\wedge\text{E.}_2) \qquad p_2 \to r}{r}\,(\to\text{E.})}{q \wedge r}\,(\wedge\text{I.})$$

We can now see very clearly that this proof proceeds by first eliminating the conjunctions in the conjunctive premiss to obtain the antecedent of the conditional premisses, which allows us to eliminate the conditionals to get to their consequents. Finally, we introduce the conjunction of these consequents.

Now, if there is an elimination rule for the conditional, one would assume that there also is an introduction rule. Indeed there is. It is the formalised version of a proof method we have been using all this time already: Conditional Proof.

Conditional Proof

To prove a claim like *if A, then B*, assume that A is the case and prove from this assumption that B.

This tells us how the formal inference rule for the introduction of a conditional should look like. To wit: one is entitled to infer $A \to B$ if one can prove B from the assumption A. We write this as follows.

$$(\to\text{I.})^i\ \dfrac{\begin{matrix}[A]^i\\ \vdots\\ B\end{matrix}}{A\to B}$$

This notation is to be understood as follows. The square brackets [ ] are used to mark formulae as *assumed.* To keep track of our assumptions we label each with a different natural number (an *index* i). The dots say that there is any proof tree between $[A]^i$ and B.

For example, we can now prove that $(p \wedge q) \to r$ entails $p \to (q \to r)$.

$$\dfrac{\dfrac{\dfrac{\dfrac{[p]^1 \qquad [q]^2}{p\wedge q}\,(\wedge\text{I.}) \qquad (p\wedge q)\to r}{r}\,(\to\text{E.})}{(q\to r)}\,(\to\text{I.})^2}{p\to(q\to r)}\,(\to\text{I.})^1$$

When we "use" an assumption to introduce a conditional, we mark its index in the step where we use (→I.). This is known as *discharging* the assumption.

After an assumption is discharged, it may not be used again. The following is a wrong proof.

$$\dfrac{\dfrac{\dfrac{[p\wedge q]^1}{p}\,(\wedge\text{E.}_1)}{(p\wedge q)\to p}\,(\to\text{I.})^1 \qquad \dfrac{[p\wedge q]^1}{q}\,(\wedge\text{E.}_2)}{((p\wedge q)\to p)\wedge q}$$

This is not a proper proof tree because the assumption with index 1 is used again after it is discharged by (→I.).

However, until an assumption is discharged it may be used arbitrarily often (like premisses). This allows us to prove the commutativity of conjunction.

$$\dfrac{\dfrac{\dfrac{[p\wedge q]^1}{q}\,(\wedge\text{E.}_2) \qquad \dfrac{[p\wedge q]^1}{p}\,(\wedge\text{E.}_1)}{q\wedge p}\,(\wedge\text{I.})}{(p\wedge q)\to(q\wedge p)}\,(\to\text{I.})^1$$

Finally, to discharge an assumption it is not necessary to have "used" it in any way. For example, we can write a proof tree showing that $p \to (q \to p)$ as follows.

$$\dfrac{\dfrac{[p]^1 \qquad [q]^2}{q\to p}\,(\to\text{I.})^2}{p\to(q\to p))}\,(\to\text{I.})^1$$

This is known as an *empty discharge.* Very technically speaking, this tree is not a correctly formed proof. This is because the rule (→I.) has above the line a *single* formula, but in this proof we have *two* formulae above the line. The problem is that $[q]$ has to be somewhere above the line where we use (→I.) discharge it, but we are not *doing* anything with it, so it just sits there uselessly.

But we can make a correctly formed proof where there is a space for q by just "uselessly" using q to introduce a conjunction and then immediately eliminate it again.

$$\cfrac{\cfrac{\cfrac{\cfrac{[p]^1 \qquad [q]^2}{p \wedge q}\,(\wedge\text{I.})}{p}\,(\wedge\text{E.}_1)}{q \to p}\,(\to\text{I.})^2}{p \to (q \to p))}\,(\to\text{I.})^1$$

Finally, it is possible to make an assumption and immediately discharge it. For example, the following proves $p \to p$.

$$\cfrac{[p]^1}{p \to p}\,(\to\text{I.})^1$$

Now we can find the rules for the rest of our connectives. An interesting case is the rule for the elimination of disjunction, which formalises the proof method of **proof by cases**.

Proof by cases

If we know that we are in one of two cases, say A or B and want to prove C, it suffices to show that A *entails C* and B *entails C*.

Again, this is a proof method we have been already using extensively. We can now formalise it as the rule for the elimination of disjunction.

$$(\vee\text{E.})^{i,j}\ \cfrac{A \vee B \qquad \begin{matrix}[A]^i \\ \vdots \\ C\end{matrix} \qquad \begin{matrix}[B]^j \\ \vdots \\ C\end{matrix}}{C}$$

This is the formal version of the following sketch for an informal proof by cases.

> *We are in one of two cases:*
>
> - *Case 1: A. In this case .... therefore C.*
> - *Case 2: B. In this case .... therefore C.*
>
> *There are no more cases (i.e. A or B is the case). Thus C.*

The rules for the introduction of disjunction are as follows. Their intuitive justification is that when we know that A is the case, we also know that we are in one of two cases: A or anything else.

$$(\vee\text{I.}_1)\ \cfrac{A}{A \vee B} \qquad (\vee\text{I.}_2)\ \cfrac{A}{B \vee A}$$

Finally, we treat the Falsum symbol $\bot$ as the formal analogue of writing "Contradiction". That is, Falsum makes explicit that we have reached a logical dead-end. We can use this idea to write the rules for the introduction and elimination of negation.

$$(\neg\text{E.})\ \cfrac{A \qquad \neg A}{\bot} \qquad (\neg\text{I.})^i\ \cfrac{\begin{matrix}[A]^i \\ \vdots \\ \bot\end{matrix}}{\neg A}$$

The idea is that when we have a formula and its negation, we know we have reached a logical dead end. We mark this by inferring $\bot$. And if from the assumption that A we reach a logical dead end, we know that A cannot be the case, so we infer $\neg A$.

We are still missing one important proof method: *reductio*.

Reductio Ad Absurdum

To prove A, assume that A is false and derive a contradiction.

Unfortunately, we cannot claim to have formalised *reductio* by the rule for the introduction of negations. That rule tells us that if from A we get a contradiction, we can infer $\neg A$. Using this rule we will *never be able to infer a non-negated formula.* But in some *reductio* arguments this is just what we want. Thus we also need to add the following rule.

$$\begin{array}{c}[\neg A]^i\\ \vdots\end{array}$$
$$(\text{RAA})^i\ \dfrac{\bot}{A}$$

One famous example for a proof that requires RAA is the derivation of the law of excluded middle. We can show that $p \vee \neg p$ by assuming $\neg(p \vee \neg p)$ and deriving $\bot$. We need to be a little bit clever about it and play with the rules for disjunction.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{[p]^1}{p \vee \neg p}\,(\vee\text{I.}_1) \qquad [\neg(p \vee \neg p)]^2}{\bot}\,(\neg\text{E.})}{\neg p}\,(\neg\text{I.})^1}{p \vee \neg p}\,(\vee\text{I.}_2) \qquad [\neg(p \vee \neg p)]^2}{\bot}\,(\neg\text{E.})}{p \vee \neg p}\,(\text{RAA})^2$$

We now have everything we need to define the syntactic consequence relation $\vdash$. We want to say that $\Gamma \vdash A$ iff there is a proof tree that ends in A and has only members of Γ as premisses. To do so, we need to define the set of all proofs.

2.7 Syntactic Consequence

In principle, the set of *trees* containing well formed formulae at their nodes/leaves contains many trees that are not *proof trees.* The following is a tree, but not a proof tree, as it does not instantiate an inference rule.

$$\dfrac{p \to q \qquad r \vee q}{p_0}$$

So we need a definition to sort out the good trees from the bad trees. The idea is the very same as when we were defining the well-formed formulae. Amongst all the possible words over an alphabet, we distinguished the *well-formed* ones by a recursive definition. Now, amongst all the possible trees over the well formed formulae, we want to distinguish the well-formed proofs (where each step instantiates an inference rule) by a recursive definition.

The plan is to define the set of all proof trees by using the inference rules as recursive steps. We still need a base case to start a recursion, but it is easy to come by. For every formula A, the following is a proof tree.

$$A$$

This is the proof tree with conclusion A and premiss A.

For the formal definition of the set of proof trees, we will use some suggestive notation. We will use variables $\mathcal{D}$, $\mathcal{D}_1$, $\mathcal{D}_2$, ... to denote fixed but arbitrary trees ($\mathcal{D}$ for *derivation*). For example, if $\mathcal{D}_1$ is a tree with conclusion A and $\mathcal{D}_2$ is a tree with conclusion B, then the following is a tree (namely, the tree obtained by continuing the derivations $\mathcal{D}_1$ and $\mathcal{D}_2$ with the rule for conjunction introduction).

$$\frac{\mathcal{D}_1 \qquad \mathcal{D}_2}{A \land B}$$

Given a tree $\mathcal{D}$, its **premisses** are all the leaves of the tree that are not surrounded by [ ]. To make the following definition a bit easier to read, we introduce the following suggestive notations:

$\begin{array}{c} \mathcal{D} \\ A \end{array}$ denotes a tree with conclusion A.

$\begin{array}{c} A \\ \mathcal{D} \\ B \end{array}$ denotes a tree with conclusion B that has A among its premisses (it may have more premisses).

$\begin{array}{c} [A] \\ \mathcal{D} \\ B \end{array}$ denotes the tree that is obtained from $\begin{array}{c} A \\ \mathcal{D} \\ B \end{array}$ by replacing A with $[A]$.

(Very technically: and $[A]$ is assigned an index that is not yet used in $\mathcal{D}$; we omit mentioning the index in this notation.)

With this we can define the set of proof trees.

Definition 27 (Proof Trees)**.** The set of proof trees is defined by the following recursion.

Base: For every $A \in \texttt{wff}$, the one-element tree A is a proof tree.

$\land$I. If $\begin{array}{c} \mathcal{D}_1 \\ A \end{array}$ and $\begin{array}{c} \mathcal{D}_2 \\ B \end{array}$ are proof trees, then $\dfrac{\begin{array}{cc} \mathcal{D}_1 & \mathcal{D}_2 \\ A & B \end{array}}{A \land B}$ is a proof tree.

$\land$E. If $\begin{array}{c} \mathcal{D} \\ A \land B \end{array}$ is a proof tree, then $\dfrac{\begin{array}{c} \mathcal{D} \\ A \land B \end{array}}{A}$ and $\dfrac{\begin{array}{c} \mathcal{D} \\ A \land B \end{array}}{B}$ are proof trees.

$\to$I. If $\begin{array}{c} A \\ \mathcal{D} \\ B \end{array}$ is a proof tree, then $\dfrac{\begin{array}{c} [A] \\ \mathcal{D} \\ B \end{array}}{A \to B}$ is a proof tree.

$\to$E. If $\begin{array}{c}\mathcal{D}_1\\ A\to B\end{array}$ and $\begin{array}{c}\mathcal{D}_2\\ A\end{array}$ are proof trees, then $\dfrac{\begin{array}{cc}\mathcal{D}_1 & \mathcal{D}_2\\ A\to B & A\end{array}}{B}$ is a proof tree.

$\vee$I. If $\begin{array}{c}\mathcal{D}\\ A\end{array}$ is a proof tree and B is a formula, then $\dfrac{\begin{array}{c}\mathcal{D}\\ A\end{array}}{A\vee B}$ and $\dfrac{\begin{array}{c}\mathcal{D}\\ A\end{array}}{B\vee A}$ are proof trees.

$\vee$E. If $\begin{array}{c}\mathcal{D}_1\\ A\vee B\end{array}$, $\begin{array}{c}A\\ \mathcal{D}_2\\ C\end{array}$, and $\begin{array}{c}B\\ \mathcal{D}_3\\ C\end{array}$ are proof trees, then $\dfrac{\begin{array}{ccc} & [A] & [B]\\ \mathcal{D}_1 & \mathcal{D}_2 & \mathcal{D}_3\\ A\vee B & C & C\end{array}}{C}$ is a proof tree.

$\neg$I. If $\begin{array}{c}A\\ \mathcal{D}\\ \bot\end{array}$ is a proof tree, then $\dfrac{\begin{array}{c}[A]\\ \mathcal{D}\\ \bot\end{array}}{\neg A}$ is a proof tree.

$\neg$E. If $\begin{array}{c}\mathcal{D}_1\\ A\end{array}$ and $\begin{array}{c}\mathcal{D}_2\\ \neg A\end{array}$ are proof trees, then $\dfrac{\begin{array}{cc}\mathcal{D}_1 & \mathcal{D}_2\\ A & \neg A\end{array}}{\bot}$ is a proof tree.

RAA If $\begin{array}{c}\neg A\\ \mathcal{D}\\ \bot\end{array}$ is a proof tree, then $\dfrac{\begin{array}{c}[\neg A]\\ \mathcal{D}\\ \bot\end{array}}{A}$ is a proof tree.

The set of proof trees is the set of correct formal proofs. This suffices to define syntactic consequence.

Definition 28. Let Γ be a set of wff and A be a wff. $\Gamma \vdash A$ iff$_{\text{Def}}$ there is a proof tree $\begin{array}{c}\mathcal{D}\\ A\end{array}$ such that all its premisses are members of Γ.

We use the same notational conventions as we used for semantic consequence. That is, instead of $\emptyset \vdash A$ we just write $\vdash A$ and when $\Gamma = \{A_0, ..., A_{n-1}\}$ is finite, we sometimes write $A_0, ..., A_{n-1} \vdash A$ instead of $\{A_0, ..., A_{n-1}\} \vdash A$.

We can now also obtain a syntactic analogoue to our alternative characterisation of semantic consequence. Define the following property of sets of premisses.

- A set of formulae Γ is called **inconsistent** iff$_{\text{Def}}$ $\Gamma \vdash \bot$. If Γ is not inconsistent, it is **consistent**.

Then the following is the case.

Theorem 2.14. *For all formulae A and sets of formulae Γ: $\Gamma \vdash A$ iff $\Gamma \cup \{\neg A\}$ is inconsistent.*

Proof. Fix arbitrary A and Γ.

$\Rightarrow$ Left-to-right: Assume that $\Gamma \vdash A$ and show that $\Gamma \cup \{\neg A\}$ is inconsistent. If $\Gamma \vdash A$ there is a proof tree as in (1) with premisses from Γ. But then (2) is a proof tree with premisses from $\Gamma \cup \{\neg A\}$. This means that $\Gamma \cup \{\neg A\} \vdash \bot$, i.e. $\Gamma \cup \{\neg A\}$ is inconsistent.

$$(1)\ \begin{array}{c}\mathcal{D}\\ A\end{array} \qquad (2)\ \dfrac{\begin{array}{c}\mathcal{D}\\ A\end{array} \qquad \neg A}{\bot}\,(\neg\mathrm{E})$$

$\Leftarrow$ Right-to-left: Assume that $\Gamma \cup \{\neg A\}$ is inconsistent and show that $\Gamma \vdash A$. If $\Gamma \cup \{\neg A\}$, we may distinguish two cases.

Case 1. $\neg A$ is a premiss in the proof of $\bot$ from $\Gamma \cup \{\neg A\}$. Then there is a proof tree as in (3) with premisses from $\Gamma \cup \{\neg A\}$ (only the premiss $\neg A$ is made explicit). But then (4) is a proof tree with premisses from Γ.

$$(3)\ \begin{array}{c}\neg A\\ \mathcal{D}\\ \bot\end{array} \qquad (4)\ \dfrac{\begin{array}{c}[\neg A]^1\\ \mathcal{D}\\ \bot\end{array}}{A}\,(\mathrm{RAA})^1$$

Case 2. $\neg A$ is not a premiss in the proof of $\bot$ from $\Gamma \cup \{\neg A\}$ (i.e. already Γ is inconsistent). Then there is a proof tree $\mathcal{D}$ that ends in $\bot$ with premisses from Γ. This means we can construct the following proof tree with premisses from Γ.

$$\dfrac{\dfrac{\dfrac{\begin{array}{c}\mathcal{D}\\ \bot\end{array} \qquad [\neg A]^1}{\bot \wedge \neg A}\,(\wedge\mathrm{I.})}{\bot}\,(\wedge\mathrm{E.}_1)}{A}\,(\mathrm{RAA})^1$$

In either case, $\Gamma \vdash A$. □

We also find syntactic analogoues to the semantic notions of a formula being a tautology and the corresponding partial characterisation of semantic consequence.

- A formula A is called a **theorem** iff$_{\text{Def}} \vdash A$.

Then the following is the case.

Theorem 2.15. *Let A be a formula and $\{A_0, ..., A_{n-1}\}$ be a finite set of formulae. Then it is the case that: $A_0, ..., A_{n-1} \vdash A$ iff $(\bigwedge_{i<n} A_i) \to A$ is a theorem.*

Proof. We show this by induction on n.

- Base. $n = 1$. Left-to-right: Assume $A_0 \vdash A$. This means there is a proof tree as in (1) with no premisses except A_0. By the recursive step for ($\to$I.) in Definition 27, this means that (2) is a proof tree. Note that (2) has no premisses, so by definition, $\vdash A_0 \to A$.

$$(1)\ \begin{array}{c}A_0\\ \mathcal{D}\\ A\end{array} \qquad (2)\ \dfrac{\begin{array}{c}[A_0]\\ \mathcal{D}\\ A\end{array}}{A_0 \to A}$$

Right-to-left: Assume $A_0 \to A$ is a theorem. This means there is a proof tree as in (3) with no premisses. By the base step in the definition of proof trees, (4) is a proof tree. Thus by the recursive step for ($\to$E.) in Definition 27, (5) is a proof tree. Its only premiss is A_0, so it shows that $A_0 \vdash A$.

$$(3)\ \begin{array}{c}\mathcal{D}\\ A_0 \to A\end{array} \qquad (4)\ A_0 \qquad (5)\ \dfrac{\begin{array}{c}\mathcal{D}\\ A_0 \to A\end{array} \qquad A_0}{A}$$

- Inductive step. Induction hypothesis: Assume that for a fixed but arbitrary n it is the case that for all formulae A and sets of formulae with n members $\{A_0, ..., A_{n-1}\}$, it is the case that $A_0, ..., A_{n-1} \vdash A$ iff $(\bigwedge_{i<n} A_i) \to A$ is a theorem.

  It is to show that for all formulae B and sets of formulae with $n + 1$ members $\{B_0, ..., B_{n-1}, B_n\}$, it is the case that $B_0, ..., B_{n-1}, B_n \vdash B$ iff $(\bigwedge_{i<n+1} B_i) \to B$ is a theorem.

  - Left-to-right: Fix B and $\{B_0, ..., B_{n-1}, B_n\}$ and assume that $B_0, ..., B_{n-1}, B_n \vdash B$. Show that $(\bigwedge_{i<n+1} B_i) \to B$ is a theorem.

    By (→I.), $B_0, ..., B_{n-1}, B_n \vdash B$ entails that $B_0, ..., B_{n-1} \vdash B_n \to B$. We can apply the induction hypothesis to conclude that $(\bigwedge_{i<n} B_i) \to (B_n \to B)$ is a theorem.

    Note that we phrased the IH for *all* A, so A in the IH need not be the same as our B here and the A_i need not be our B_i. In particular, we use the IH for $A = B_n \to B$.

    Now, $(\bigwedge_{i<n} B_i) \to (B_n \to B)$ being a theorem means that there is a proof tree as the following with no premisses.

    $$\begin{array}{c}\mathcal{D}\\ (\bigwedge_{i<n} B_i) \to (B_n \to B)\end{array}$$

    This means that the following is a proof tree with no premisses.

    $$\begin{prooftree}
    \AxiomC{$\mathcal{D}$}
    \noLine
    \UnaryInfC{$(\bigwedge_{i<n} B_i) \to (B_n \to B)$}
    \AxiomC{$[(\bigwedge_{i<n} B_i) \land B_n)]^1$}
    \RightLabel{($\land$E.$_1$)}
    \UnaryInfC{$\bigwedge_{i<n} B_i$}
    \RightLabel{($\to$E.)}
    \BinaryInfC{$B_n \to B$}
    \AxiomC{$[(\bigwedge_{i<n} B_i) \land B_n)]^1$}
    \RightLabel{($\land$E.$_2$)}
    \UnaryInfC{B_n}
    \RightLabel{($\to$E.)}
    \BinaryInfC{B}
    \RightLabel{($\to$I.)^1}
    \UnaryInfC{$(\bigwedge_{i<n} B_i \land B_n) \to B$}
    \end{prooftree}$$

    But $(\bigwedge_{i<n} B_i \land B_n) = \bigwedge_{i<n+1} B_i$. So this shows that $\vdash (\bigwedge_{i<n+1} B_i) \to B$. This was to show.

  - Right-to-left: Fix B and $\{B_0, ..., B_{n-1}, B_n\}$ and assume that $(\bigwedge_{i<n+1} B_i) \to B$ is a theorem. It is to show that $B_0, ..., B_{n-1}, B_n \vdash B$.

    We now use a trick to find something to apply the IH to: let $C_0 = B_0 \land B_1$ and for all i with $0 < i < n$, let $C_i = B_{i+1}$. Note that $\bigwedge_{i<n+1} B_i = \bigwedge_{i<n} C_i$ (literally, it is the exact same formula). Thus we have that $(\bigwedge_{i<n} C_i) \to B$ is a theorem by assumption.

    By our IH, it follows that $C_0, ..., C_{n-1} \vdash B$. This means that $B_0 \land B_1, B_2, ..., B_n \vdash B$. Using (∧E.) we can re-write the proof tree to a proof tree showing that $B_0, B_1, B_2, ..., B_n \vdash B$. This was to show.

This concludes the induction. □

For syntactic consequence, we can immediately prove compactness.

Theorem 2.16 (Syntactic Compactness). $\Gamma \vdash A$ *iff there is a finite subset* $\Gamma' \subseteq \Gamma$ *with* $\Gamma' \vdash A$.

Proof. Every proof tree can only have finitely many premisses. Thus if $\Gamma \vdash A$, then let Γ' be the premisses occurring in the proof tree showing that $\Gamma \vdash A$. By definition, by the very same proof tree, $\Gamma' \vdash A$ and by construction Γ' is finite.

Conversely, if there is a finite $\Gamma' \subseteq \Gamma$ with $\Gamma' \vdash A$ then there is a proof tree with conclusion A and premisses from Γ'. This is also a proof tree with conclusion A and premisses from Γ. □

But again, we cannot express 'there is a finite subset of the premisses' in the object language, so there is no single formula whose theoremhood is equivalent to $\Gamma \vdash A$ if Γ may be infinite.

2.8 Derived Rules

There are a few proof methods that we have not yet captured in our calculus. For example:

Proof by Contraposition

To prove that if A, then B, prove that if B is false, then A is false.

This could be expressed in the following inference rule.

$$(\text{Contraposition})\ \frac{\neg B \to \neg A}{A \to B}$$

It is not necessary to add such a rule to our calculus, because we can *derive* it. First consider how to prove Contraposition for a concrete formula. The following proof shows that $\neg q \to \neg p$ entails $p \to q$.

$$\dfrac{\dfrac{\dfrac{\dfrac{\neg q \to \neg p \qquad [\neg q]^2}{\neg p}\,(\to\text{E.}) \qquad [p]^1}{\bot}\,(\neg\text{E.})}{q}\,(\text{RAA})^2}{p \to q}\,(\to\text{I})^1$$

But this is not quite what we wanted: the rule of (Contraposition) is supposed to be *schematic*. But recall that all our proof rules were schematic as well. Thus if we replace p with any A and q with any B in the above proof, we still have a valid proof tree. That is, **because all proof rules are schematic, proofs themselves are schematic**. Thus, the above proof tree suffices to show that it is always correct to use the rule (Contraposition) in a proof. We call (Contraposition) a **derived rule** because we can derive its correctness from the basic proof rules.

The other direction of contraposition is sometimes useful as well.

$$(\text{Contraposition}')\ \frac{A \to B}{\neg B \to \neg A}$$

The proof is very similar to the one above, but uses (¬I.) instead of (RAA).

$$\dfrac{\dfrac{\dfrac{\dfrac{q \to p \qquad [q]^2}{p}\,(\to\text{E.}) \qquad [\neg p]^1}{\bot}\,(\neg\text{E.})}{\neg q}\,(\neg\text{I})^2}{\neg p \to \neg q}\,(\to\text{I})^1$$

Some further useful derived rules are double-negation elimination (DNE) and introduction (DNI).

$$(\text{DNE})\ \frac{\neg\neg A}{A} \qquad (\text{DNI})\ \frac{A}{\neg\neg A}$$

The proofs are as follows:

$$\dfrac{\dfrac{[\neg A]^1 \qquad \neg\neg A}{\bot}\,{\scriptstyle(\neg\text{E.})}}{A}\,{\scriptstyle(\text{RAA})^1} \qquad \dfrac{\dfrac{A \qquad [\neg A]^1}{\bot}\,{\scriptstyle(\neg\text{E.})}}{\neg\neg A}\,{\scriptstyle(\neg\text{I})^1}$$

Note that only the rules for negation are requires to derive the converse of contraposition and the double-negation introduction rule. But for Contrapositoin and double-negation elimination we require Reductio ad Absurdum. To the intuitionist logician, all of RAA, Contraposition and DNE are suspect.

We can now also see that the Natural Deduction calculus has more connectives than it needs. For example, we could remove the rules for $\vee$ and instead treat $A \vee B$ as an abbreviation of $\neg(\neg A \wedge \neg B)$. To show that we did not need $\vee$, we need to derive the rules for disjunction. That is, in the calculus without disjunction, we need to derive the following.

$$(\vee\text{I.}_1{}^*)\ \frac{A}{\neg(\neg A \wedge \neg B)} \qquad (\vee\text{I.}_2{}^*)\ \frac{A}{\neg(\neg B \wedge \neg A)} \qquad (\vee\text{E.}^*)^{i,j}\ \frac{\neg(\neg A \wedge \neg B) \qquad \begin{array}{c}[A]^i\\ \vdots\\ C\end{array} \qquad \begin{array}{c}[B]^j\\ \vdots\\ C\end{array}}{C}$$

Proof that the Introduction rules are derivable; this is the proof for ($\vee$I.$_1{}^*$), as ($\vee$I.$_2{}^*$) is analogous.

$$\dfrac{\dfrac{A \qquad \dfrac{[\neg A \wedge \neg B]^1}{\neg A}\,{\scriptstyle(\wedge\text{E}_1)}}{\bot}\,{\scriptstyle(\neg\text{E})}}{\neg(\neg A \wedge \neg B)}\,{\scriptstyle(\neg\text{I})^1}$$

The derivation of ($\vee$E.*) is as follows.

$$\dfrac{\dfrac{\neg(\neg A \wedge \neg B) \qquad \dfrac{\dfrac{\dfrac{\dfrac{\begin{array}{c}[A]^i\\ \vdots\\ C\end{array}}{A \to C}\,{\scriptstyle(\to\text{I})^i}}{\neg C \to \neg A}\,{\scriptstyle(\text{Contraposition}')} \qquad [\neg C]^1}{\neg A}\,{\scriptstyle(\to\text{E.})} \qquad \dfrac{\dfrac{\dfrac{\begin{array}{c}[B]^j\\ \vdots\\ C\end{array}}{B \to C}\,{\scriptstyle(\to\text{I})^j}}{\neg C \to \neg B}\,{\scriptstyle(\text{Contraposition}')} \qquad [\neg C]^1}{\neg B}\,{\scriptstyle(\to\text{E.})}}{\neg A \wedge \neg B}\,{\scriptstyle(\wedge\text{I.})}}{\bot}\,{\scriptstyle(\neg\text{E.})}}{C}\,{\scriptstyle(\text{RAA})^1}$$

It is *usually* the case that if you have a propositional basis, then the Natural Deduction rules for the connectives in the basis allow you to prove the rules for the abbreviated connectives as derived rules. But this depends on how exactly the rules are phrased—and in any case, needs to be *proved* for every basis. For example, since we have here defined $\neg$ by using $\bot$, we cannot use the basis $\{\neg, \vee, \wedge\}$, but require $\{\neg, \vee, \wedge, \bot\}$. It is, however, possible to phrase rules for negation without $\bot$.

We will now turn to discussing a different calculus in which we can use a very small set of rules and connectives.

2.9 The Hilbert Calculus

The Natural Deduction calculus might succeed in formalising our 'natural' proof methods, but the proof trees it produces do not look very natural. Perhaps one would prefer proofs just to be sequences of formulae (like natural proofs appear to be). We can have this when we use the *Hilbert calculus* instead of the Natural Deduction one.

The idea behind the Hilbert calculus is that we only have *one* fundamental inference rule: *modus ponens.*

$$\text{Modus Ponens}\ \frac{A \to B \qquad A}{B}$$

This single inference rule is supplemented with a number of *logical axioms*, which are schemata for formulae. The following are *Frege's axioms* for the propositional calculus. (Some of these are redundant. For example, F3 follows from the others, and if we were to replace F4 with its converse, we could derive F5 and F6.)

F1 $A \to (B \to A)$.

F2 $(A \to (B \to C)) \to ((A \to B) \to (A \to C))$.

F3 $(A \to (B \to C)) \to (B \to (A \to C))$.

F4 $(A \to B) \to (\neg B \to \neg A)$.

F5 $\neg\neg A \to A$.

F6 $A \to \neg\neg A$.

And this is it. It has become a bit of a sport among logicians to find the smallest sets of logical axioms for the propositional calculus (single-axiom systems are known, but are very annoying to use).

We can now define syntactic consequence as follows.

Definition 29. $\Gamma \vdash^H A$ iff there is an ordered set $\langle A_0, ..., A_n \rangle$ such that $A_n = A$ and for every $i \leq n$ one of the following is the case:

- A_i is an instance of a logical axiom.
- A_i is a member of Γ.
- There are $j < i$ and $k < i$ such that $\frac{A_j \qquad A_k}{A_i}$ is an instance of Modus Ponens.

Because $\{\neg, \to\}$ is a propositional base (exercise), we can treat $\wedge$, $\vee$ and $\bot$ as abbreviations to obtain a syntactic consequence relation for our full language.

The Hilbert calculus has some advantages. It is typically easier to define the behaviour of a new logical operator in terms of axioms than it is to define it in terms of rules for its Introduction and Elimination. This is why modal logics are typically stated in terms of Hilbert calculi.

In addition, because it is very minimal, it is easier to prove things *about* the Hilbert calculus (as we will see). But proving things *within* the Hilbert calculus is much harder—and indeed, less natural—than in the

Natural Deduction calculus. To even get started proving things, it is very useful to establish some derived rules. The following is the *hypothetical syllogism*.

$$\text{(HS)}\ \frac{A \to B \qquad B \to C}{A \to C}$$

As an example, we will prove that this is a derived rule here. To show this, first show that $\vdash^H (q \to r) \to ((p \to q) \to (p \to r))$. This is established by the following sequence of formulae.

1.	$(p \to (q \to r)) \to ((p \to q) \to (p \to r))$	Instance of F2
2.	$((p \to (q \to r)) \to ((p \to q) \to (p \to r)))$ $\to ((q \to r) \to ((p \to (q \to r)) \to ((p \to q) \to (p \to r))))$	Instance of F1
3.	$(q \to r) \to ((p \to (q \to r)) \to ((p \to q) \to (p \to r)))$	Modus Ponens 1,2
4.	$((q \to r) \to ((p \to (q \to r)) \to ((p \to q) \to (p \to r))))$ $\to (((q \to r) \to (p \to (q \to r))) \to ((q \to r) \to ((p \to q) \to (p \to r))))$	Instance of F2
5.	$((q \to r) \to (p \to (q \to r))) \to ((q \to r) \to ((p \to q) \to (p \to r)))$	Modus Ponens 3, 4
6.	$(q \to r) \to (p \to (q \to r))$	Instance of F1
7.	$(q \to r) \to ((p \to q) \to (q \to r))$	Modus Ponens 5,6.

This establishes $\vdash^H (q \to r) \to ((p \to q) \to (p \to r))$. The derived rule (HS) then follows from two applications of Modus Ponens. To wit, we can write the following *schema* for a proof that shows that for all A, B and C, it is the case that $A \to B, B \to C \vdash^H A \to C$.

1.	$(B \to C) \to ((A \to B) \to (A \to C))$	by the proof above
2.	$A \to B$	premiss
3.	$B \to C$	premiss
4.	$(A \to B) \to (A \to C)$	Modus Ponens, 1,3
5.	$A \to C$	Modus Ponens, 2,4

The following is arguably the most important result about the Hilbert calculus. It will make our lives a lot easier.

Theorem 2.17 (Deduction Theorem)**.** *Let A and B be formulae and Γ be a set of formulae. If $\Gamma \cup \{A\} \vdash^H B$, then $\Gamma \vdash A \to B$.*

Proof. We prove this by induction on the length n on the proof of B from $\Gamma \cup \{A\}$.

- Base case: $n = 1$. Recall that a proof of B in the Hilbert calculus is an ordered set of formulae where the last entry is B. So a proof of B that has length 1 is $\langle B \rangle$. This means that B must be a member of $\Gamma \cup \{A\}$ or B is an instance of a logical axiom. Thus there are three cases:
  - Case 1: $B = A$. Then we need to show that $\Gamma \vdash A \to A$. This follows from an exercise.
  - Case 2: $B \in \Gamma$. The following proof shows $\Gamma \vdash A \to B$.

    1. $B \to (A \to B)$ Instance of F2.
    2. B Member of Γ
    3. $A \to B$ Modus Ponens 1,2

  - Case 3: B is a logical axiom. The following proof shows $\Gamma \vdash A \to B$.

    1. $B \to (A \to B)$ Instance of F2.
    2. B Instance of a logical axiom
    3. $A \to B$ Modus Ponens 1,2

- Inductive step. Let n be fixed but arbitrary and assume the induction hypothesis: for all Γ, A and B such that $\Gamma \cup \{A\} \vdash^H B$ by a proof of length n or less, $\Gamma \vdash^H A \to B$. Show that for all Γ, A and B be such that $\Gamma \cup \{A\} \vdash^H B$ by a proof of length $n+1$ or less, $\Gamma \vdash^H A \to B$.

  So let Γ, A and B be such that $\Gamma \cup \{A\} \vdash^H B$ by a proof of length $n+1$ or less. We can distinguish two cases:

  - Case 1. $B = A$ or $B \in \Gamma$ or B is a logical axiom. Then proceed as in the base case.
  - Case 2. B follows from an application of Modus Ponens. This means that before B in the proof there are formulae C and $C \to B$. The proofs of C and $C \to B$ have length n or shorter, so we can apply the induction hypothesis. Hence $\Gamma \vdash A \to C$ and $\Gamma \vdash A \to (C \to B)$.

    Note that the following is an instance of F2: $(A \to (C \to B)) \to ((A \to C) \to (A \to B)))$. We can construct a proof showing that $\Gamma \vdash A \to B$ as follows.

    1. [copy of the proof that $\Gamma \vdash A \to (C \to B)$]
    2. $(A \to (C \to B)) \to ((A \to C) \to (A \to B))$ Instance of F2
    3. $(A \to C) \to (A \to B)$ Modus Ponens 1,2
    4. [copy of the proof that $\Gamma \vdash A \to C$]
    5. $A \to B$ Modus Ponens 4,5

    Very technically, this can also be put as follows: from the induction hypothesis, we know that there is a proof P_1 (i.e. an ordered set of formulae) whose last entry is $A \to (C \to B)$ and that there is a proof P_2 whose last entry is $A \to C$. Both P_1 and P_2 contain only premisses from Γ. Then the following ordered set of formulae is a proof of $A \to B$ from Γ:

    $$P_1^\frown \langle (A \to (C \to B) \to ((A \to C) \to (A \to B))) \rangle^\frown \langle (A \to C) \to (A \to B) \rangle^\frown P_2^\frown \langle A \to B \rangle.$$

    Thus $\Gamma \vdash A \to B$.

This concluses the induction. □

The proof of the Deduction Theorem showcases how easy it is to show things *about* the Hilbert calculus. If we need to do a case distinction on steps in proofs, there are only three options. Usually the first two (member of the premisses or logical axioms) are easily dealt with, so we only need to prove things about Modus Ponens. That keeps these proofs very straightforward.

We can give now the following proof (in the meta-language) that $\vdash^H (q \to r) \to ((p \to q) \to (p \to r))$ without explicitly constructing an object-language argument (as we did above).

- First show that $q \to r, p \to q, p \vdash^H r$.

1.	$p \to q$	Premiss
2.	p	Premiss
3.	q	Modus Ponens 1,2
4.	$q \to r$	Premiss
5.	r	Modus Ponens 3,4

- By Deduction, $q \to r, p \to q \vdash^H p \to r$
- By Deduction, $q \to r, \vdash^H (p \to q) \to (p \to r)$
- By Deduction, $\vdash^H (q \to r) \to ((p \to q) \to (p \to r))$

Thus, the Deduction theorem allows us to prove *that* the Hilbert calculus proves certain things without stating an explicit formal proof. This is why the Deduction theorem is sometimes called a *metatheorem*: it is a meta-language result *about* which proofs are possible in the object language.

Here is another useful result (a version of the Explosion rule).

Theorem 2.18 (Explosion)**.** $A \vdash^H \neg A \to B$

Proof. The formal proof is as follows.

1.	$A \to (\neg B \to A)$	Instance of F1
2.	A	Premiss
3.	$\neg B \to A$	Modus Ponens 1,2
4.	$(\neg B \to A) \to (\neg A \to \neg\neg B)$	Instance of F4
5.	$\neg A \to \neg\neg B$	Modus Ponens 3,4
6.	$\neg\neg B \to B$	Instance of F6
7.	$\neg A \to B$	Hypothetical Syllogism, 5,6

□

From this, we can obtain a rather weak version of Negation Introduction in the Hilbert calculus.

Theorem 2.19. $\vdash^H (A \to \neg A) \to \neg A.$

Proof. Show $(A \to \neg A) \vdash^H \neg A$ and apply the Deduction theorem.

1.	$A \to (\neg A \to \neg(A \to A))$	Previous theorem
2.	$(A \to (\neg A \to \neg(A \to A))) \to ((A \to \neg A) \to (A \to \neg(A \to A)))$	Instance of F2
3.	$(A \to \neg A) \to (A \to \neg(A \to A))$	Modus Ponens 1,2
4.	$(A \to \neg A)$	Premiss
5.	$A \to \neg(A \to A)$	Modus Ponens 3,4
6.	$(A \to \neg(A \to A)) \to (\neg\neg(A \to A) \to \neg A)$	Instance of F4
7.	$\neg\neg(A \to A) \to \neg A$	Modus Ponens 5,6
8.	$(A \to A) \to \neg\neg(A \to A)$	Instance of F6
9.	$A \to A$	from a previous result
10.	$\neg\neg(A \to A)$	Modus Ponens 8,9
11.	$\neg A$	Modus Ponens 7,10 □

From this we get a proper version of the Negation Introduction rule.

Theorem 2.20. $\vdash^H (A \to B) \to ((A \to \neg B) \to \neg A))$.

Proof. The following proof shows that $A \to B, A \to \neg B \vdash^H \neg A$. Then apply the Deduction theorem.

1.	$(A \to \neg B) \to (\neg\neg B \to \neg A)$	Instance of F4
2.	$A \to \neg B$	Premiss
3.	$\neg\neg B \to \neg A$	Modus Ponens 1,2
4.	$B \to \neg\neg B$	Instance of F6
5.	$B \to \neg A$	Hypothetical Syllogism 3,4
6.	$A \to B$	Premiss
7.	$A \to \neg A$	Hypothetical Syllogism 5,6
8.	$\neg A$	by the previous th. □

2.10 Soundness

We want to show now that whenever $\Gamma \vdash A$, then also $\Gamma \models A$. We first make the Natural Deduction calculus a bit more like the Hilbert calculus, so that we have to deal with fewer proof rules. Let $\bot$, $\wedge$ and $\vee$ as abbreviations in the usual way. That is, $A \wedge B$ abbreviates $\neg(A \to \neg B)$, $A \vee B$ abbreviates $\neg(\neg A \wedge \neg B)$ and $\bot$ abbreviates $p \wedge \neg p$.

Theorem 2.21. *The following rules suffice to derive all other rules of Natural Deduction.*

$$(\to\text{I.})^i\ \frac{\begin{array}{c}[A]^i\\ \vdots\\ B\end{array}}{A \to B} \qquad (\to\text{E.})\ \frac{A \to B \qquad A}{B} \qquad (\text{NI})\ \frac{A \to B \qquad A \to \neg B}{\neg A} \qquad (\text{DNE})\ \frac{\neg\neg A}{A} \qquad (\text{E})\ \frac{A \qquad B}{A}$$

Proof. Begin with the rules for conjunction. The following proof tree shows that ($\wedge$I.) is derivable.

$$\dfrac{\dfrac{\dfrac{[A \to \neg B]^1 \qquad A}{\neg B}\,(\to\text{E.})}{(A \to \neg B) \to \neg B}\,(\to\text{I.})^1 \qquad \dfrac{\dfrac{B \qquad [A \to \neg B]^2}{B}\,(\text{E})}{(A \to \neg B) \to B}\,(\to\text{I.})^2}{\neg(A \to \neg B)}\,(\text{NI})$$

The following proof tree shows that $(\wedge\text{E.}_1)$ is derivable.

$$\dfrac{\dfrac{\dfrac{\dfrac{\neg(A \to \neg B) \qquad [\neg A]^1}{\neg(A \to \neg B)}\,(\text{E})}{\neg A \to \neg(A \to \neg B)}\,(\to\text{I.})^1 \qquad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{[\neg A]^2 \qquad [B]^3}{\neg A}\,(\text{E})}{B \to \neg A}\,(\to\text{I.})^3 \qquad \dfrac{\dfrac{[A]^4 \qquad [B]^5}{A}\,(\text{E})}{B \to A}\,(\to\text{I.})^5}{\neg B}\,(\text{NI})}{A \to \neg B}\,(\to\text{I.})^4}{\neg A \to (A \to \neg B)}\,(\to\text{I.})^2}{\neg\neg A}\,(\text{NI})}{A}\,(\text{DNE})$$

The next one shows the derivation of $(\wedge\text{E.}_2)$.

$$\dfrac{\dfrac{\dfrac{\dfrac{\neg(A \to \neg B) \qquad [\neg B]^1}{\neg(A \to \neg B)}\,(\text{E})}{\neg B \to \neg(A \to \neg B)}\,(\to\text{I.})^1 \qquad \dfrac{\dfrac{\dfrac{[\neg B]^2 \qquad [A]^3}{\neg B}\,(\text{E})}{A \to \neg B}\,(\to\text{I.})^3}{\neg B \to (A \to \neg B)}\,(\to\text{I.})^2}{\neg\neg B}\,(\text{NI})}{B}\,(\text{DNE})$$

The next proof trees show the derivability of (¬I.) and (¬E.).

$$\dfrac{\dfrac{\dfrac{\begin{matrix}[A]^1\\ \vdots\\ \bot\end{matrix}}{p}\,(\wedge\text{E.}_1)}{A \to p}\,(\to\text{I.})^1 \qquad \dfrac{\dfrac{\begin{matrix}[A]^2\\ \vdots\\ \bot\end{matrix}}{\neg p}\,(\wedge\text{E.}_2)}{A \to \neg p}\,(\to\text{I.})^1}{\neg A}\,(\text{NI}) \qquad \dfrac{\dfrac{\dfrac{\dfrac{A \quad [\neg\bot]^1}{A}\,(\text{E})}{\neg\bot \to A}\,(\to\text{I.})^1 \qquad \dfrac{\dfrac{\neg A \quad [\neg\bot]^2}{\neg A}\,(\text{E})}{\neg\bot \to \neg A}\,(\to\text{I.})^2}{\neg\neg\bot}\,(\text{NI})}{\bot}\,(\text{DNE})$$

Finally, we derive (RAA) in the following proof tree.

$$\dfrac{\dfrac{\dfrac{\dfrac{\begin{matrix}[\neg A]^1\\ \vdots\\ \bot\end{matrix}}{p}\,(\wedge\text{E.}_1)}{\neg A \to p}\,(\to\text{I.})^1 \qquad \dfrac{\dfrac{\begin{matrix}[\neg A]^2\\ \vdots\\ \bot\end{matrix}}{\neg p}\,(\wedge\text{E.}_2)}{\neg A \to \neg p}\,(\to\text{I.})^1}{\neg\neg A}\,(\text{NI})}{A}\,(\text{DNE})$$

The rules for → need not be derived and we have already seen above how to derive the rules for ∨. □

We will use this reduced calculus to prove things *about* Natural Deduction. We now have a smaller, more Hilbert-like version of Natural Deduction for these purposes, and our old, large version of Natural Deduction to prove things *in* the proof theory nicely. We will now just talk about the Natural Deduction calculus and sometimes mean either one of these versions, whichever is more convenient for the task at hand.

The next useful thing we can do is define the *length* of a Natural Deduction proof (intuitively, this is the number of steps in a proof).

Definition 30. The length $\texttt{len}(\mathcal{D})$ of a proof tree $\mathcal{D}$ is defined by the following recursion.

- Base: if $\mathcal{D}$ is a proof tree consisting of a single formula, then $\texttt{len}(\mathcal{D}) = 1$.
- Recursive steps:

– If $\texttt{len}(\begin{array}{c} A \\ \mathcal{D} \\ B \end{array}) = n$, then define $\texttt{len}(\dfrac{\begin{array}{c} [A] \\ \mathcal{D} \\ B \end{array}}{A \to B}) = n + 1$.

– If $\texttt{len}(\begin{array}{c} \mathcal{D}_1 \\ A \to B \end{array}) = n$ and $\texttt{len}(\begin{array}{c} \mathcal{D}_2 \\ A \end{array}) = m$, then define $\texttt{len}(\dfrac{\begin{array}{c} \mathcal{D}_1 \\ A \to B \end{array} \quad \begin{array}{c} \mathcal{D}_2 \\ A \end{array}}{B}) = n + m + 1$.

– If $\texttt{len}(\begin{array}{c} \mathcal{D}_1 \\ A \to B \end{array}) = n$ and $\texttt{len}(\begin{array}{c} \mathcal{D}_2 \\ A \to \neg B \end{array}) = m$, then $\texttt{len}(\dfrac{\begin{array}{c} \mathcal{D}_1 \\ A \to B \end{array} \quad \begin{array}{c} \mathcal{D}_2 \\ A \to \neg B \end{array}}{\neg A}) = n + m + 1$.

– If $\texttt{len}(\begin{array}{c} \mathcal{D} \\ \neg\neg A \end{array}) = n$ then define $\texttt{len}(\dfrac{\begin{array}{c} \mathcal{D} \\ \neg\neg A \end{array}}{A}) = n + 1$.

– If $\texttt{len}(\begin{array}{c} \mathcal{D}_1 \\ A \end{array}) = n$ and $\texttt{len}(\begin{array}{c} \mathcal{D}_2 \\ B \end{array}) = m$, then $\texttt{len}(\dfrac{\begin{array}{c} \mathcal{D}_1 \\ A \end{array} \quad \begin{array}{c} \mathcal{D}_2 \\ B \end{array}}{A}) = n + m + 1$.

Now we can (surprisingly straightforwardly) prove that the Hilbert calculus is equivalent to the Natural Deduction calculus (when, in the Hilbert calculus, we treat connectives other than $\neg$ and $\to$ as abbreviations).

Theorem 2.22. *For all sets of formulae Γ and formulae A, $\Gamma \vdash^H A$ iff $\Gamma \vdash A$.*

Proof. In the Hilbert calculus, we treat the connectives other than $\to$ and $\neg$ as abbreviations. The proof is only given as a sketch, as the details are easy to work out.

$\Rightarrow$ Left-to-right. Because Natural Deduction also contains Modus Ponens, it suffices to show that all logical axioms of the Hilbert calculus are theorems of the Natural Deduction calculus. For F4, F5 and F6, we have shown this already and F1, F2 and F3 are straightforward applications of the ($\to$I.) rule.

$\Leftarrow$ Right-to-left. Show by induction on the length of proof trees that whenever $\Gamma \vdash A$, then also $\Gamma \vdash^H A$.

- Base: Suppose that $\Gamma \vdash A$ by a proof tree of length 1. Then A is a member of Γ. But the proof consisting of just A premiss is also a proof in the Hilbert calculus for $\Gamma \vdash^H A$.
- Inductive step. IH: Assume that for a fixed but arbitrary n it is the case that for all Γ and A, if $\Gamma \vdash A$ with a proof tree of length n or less, then also $\Gamma \vdash^H A$.

It is to show that for all Γ and A where $\Gamma \vdash A$ by a proof tree of length $n+1$ it is the case that $\Gamma \vdash^H A$. So let Γ and A be arbitrary such that A follows from Γ by a proof tree of length $n+1$. We can do a case distinction on the *last step* in the proof of A.

- If the last step is ($\rightarrow$I.), then $A = B \rightarrow C$ for some formulae B and C and there is a proof tree showing that $\Gamma \cup \{B\} \vdash C$. This proof tree has length n, so by the IH, $\Gamma \cup \{B\} \vdash^H C$. So by the Deduction theorem, $\Gamma \vdash^H B \rightarrow C$.
- If the last step is ($\rightarrow$E.), then there are proof trees showing that for some B, $\Gamma \vdash B \rightarrow A$ and $\Gamma \vdash B$. Both are shorter than the proof of A, so by the IH $\Gamma \vdash^H B \rightarrow A$ and $\Gamma \vdash^H B$. So by Modus Ponens in the Hilbert calculus, $\Gamma \vdash^H A$.
- If the last step is (E), there is nothing to show; if it is (DNE) proceed as in the ($\rightarrow$I.) case; if it is (NI) use Theorem 2.20 and proceed as in the ($\rightarrow$I.) case.

This concludes the induction. □

In this proof, we could have used an induction on the construction of proof trees instead of an induction on the length of proofs. This would look almost the same, except that we would need to write separate induction hypotheses for every inference rule. The definition of the length of a proof allows us to cover all the inference rules in a single inductive step. In almost all cases, induction on length is simply a more concise way to write down induction on the construction of proofs.

We now use the same technique to prove the Soundness of the Natural Deduction calculus (and hence also of the Hilbert calculus).

Theorem 2.23 (Soundness of the Propositional Calculus)**.** *For all sets of formulae Γ and formulae A, if $\Gamma \vdash A$, then $\Gamma \models A$.*

Proof. Show by induction on the length of proof trees that whenever $\Gamma \vdash A$, then also $\Gamma \models A$.

- Base: Suppose that $\Gamma \vdash A$ by a proof tree of length 1. Then A is a member of Γ. Then it is trivially the case that for all valuations $\mathcal{V}$, if all members of Γ are true given $\mathcal{V}$ then also A is true given $\mathcal{V}$.
- Inductive step. IH: Assume that for a fixed but arbitrary n it is the case that for all Γ and A, if $\Gamma \vdash A$ with a proof tree of length n or less, then also $\Gamma \models A$.

  It is to show that for all Γ and A where $\Gamma \vdash A$ by a proof tree of length $n+1$ it is the case that $\Gamma \models A$. So let Γ and A be arbitrary such that A follows from Γ by a proof tree of length $n+1$. We can do a case distinction on the last step in the proof of A.

  - If the last step is ($\rightarrow$E.), then there are proof trees showing that for some B, $\Gamma \vdash B \rightarrow A$ and $\Gamma \vdash B$. Both are shorter than the proof of A, so by the IH $\Gamma \models B \rightarrow A$ and $\Gamma \models B$. We can read off the Truth Table for $\rightarrow$ that all valuations that make B and $B \rightarrow A$ true also make A true. Hence all valuations that make all members of Γ true, make B and $B \rightarrow A$ true, so make A true. Hence $\Gamma \models A$.
  - If the last step is (NI), then for some C, $A = \neg C$ and there are proof trees showing that for some B, $\Gamma \vdash C \rightarrow B$ and $\Gamma \vdash C \rightarrow \neg B$. Both are shorter than the proof of A, so by the IH $\Gamma \models C \rightarrow B$

and $\Gamma \models C \to \neg B$. We can read off the Truth Table for → that all valuations that make $C \to B$ and $C \to \neg B$ true must make C false, so A true. Hence $\Gamma \models A$.

- If the last step is (DNE), then $\Gamma \vdash \neg\neg A$ by a proof tree of length n, so by the IH $\Gamma \models \neg\neg A$. As we had shown $\neg\neg A \approx A$, it follows that $\Gamma \models A$.
- If the last step is (E), then there are proof trees showing that for some B, $\Gamma \vdash A$ and $\Gamma \vdash B$. Both are shorter than the proof of A we are considering in this step, so by the IH $\Gamma \models A$.
- If the last step is (→I.), then $A = B \to C$ for some formulae B and C and there is a proof tree of length n showing that $\Gamma \cup \{B\} \vdash C$. By the IH, $\Gamma \cup \{B\} \models C$. Towards a *reductio*, assume that there is a valuation $\mathcal{V}$ that makes all members of Γ true, but $\mathcal{V}^*(B \to C) = 0$.

  By the Truth Table for →, it follows that $\mathcal{V}^*(B) = 1$ and $\mathcal{V}^*(C) = 0$. But this means that $\mathcal{V}$ makes all members of $\Gamma \cup \{B\}$ true and C false. This contradicts $\Gamma \cup \{B\} \models C$. By *reductio*, $\Gamma \models B \to C$.

This concludes the induction. □

Our reduction of the Natural Deduction calculus has made this proof pleasingly short. But it is instructive to consider how the proof would go for the full calculus. For example, the case for the disjunction elimination rule would go as follows.

- If the last step is (∨E.), then there are formulae B and C such that $\Gamma \vdash B \vee C$ and $\Gamma \cup \{B\} \vdash A$ and $\Gamma \cup \{C\} \vdash A$, all by by proof trees shorter than $n + 1$, it follows that $\Gamma \models B \vee C$, $\Gamma \cup \{B\} \models A$ and $\Gamma \cup \{C\} \models A$.

  By the Truth Table for ∨ and because $\Gamma \models B \vee C$, it follows that $\mathcal{V}^*(B) = 1$ or $\mathcal{V}^*(C) = 1$. If the former, then $\mathcal{V}$ makes all members of $\Gamma \cup \{B\}$ true, so it makes A true because $\Gamma \cup \{B\} \models A$. If the latter, then $\mathcal{V}$ makes all members of $\Gamma \cup \{C\}$ true, so it makes A true because $\Gamma \cup \{C\} \models A$. Thus if $\mathcal{V}$ makes all members of Γ true, it makes A true.

Note that in this proof we appeal to the *meta-language* proof method of Proof by Cases, which is exactly the proof method that is formalised in the inference rule whose Soundness we are proving here. This means that we have now established that our formalisations of our proof methods preserve truth (for our formalisation of truth)—but we have used the informal versions of these methods to show this.

This is not by accident: we had set up the Truth Tables to capture our natural use and we had set up the inference rules to capture our natural proof techniques. In a sense, the Soundness proof establishes no more than we have not made any mistakes in doing so. So, if you had worries about whether the proof methods we use in our natural mathematical proofs are valid (i.e. preserving truth), the Soundness result can do little to assuage your worries.

But then, why prove Soundness? One reason why it is useful to have a Soundness result is that it allows you to prove in a straightforward way that certain things are *not* provable from premisses.

Theorem 2.24. *Let Γ be a set of formulae and A be a formula. If there is a valuation $\mathcal{V}$ such that $\mathcal{V}^*(A) = 0$ and for all $B \in \Gamma$, $\mathcal{V}^*(B) = 1$, then $\Gamma \nvdash A$.*

Proof. Let Γ, A and $\mathcal{V}$ be arbitrary such that $\mathcal{V}^*(A) = 0$ and for all $B \in \Gamma$, $\mathcal{V}^*(B) = 1$. Then $\mathcal{V}^*(\neg A) = 1$ by the Truth Table for negation. Thus all members of $\Gamma \cup \{\neg A\}$ are true given $\mathcal{V}$. So $\Gamma \cup \{\neg A\}$ is satisfiable.

By the contrapositive of Theorem 2.13, this means that $\Gamma \not\models A$. By the contrapositive of Soundness, this means that $\Gamma \nvdash A$. □

We can galvanise this result in a proof method for proving nonprovability.

Proving Nonprovability by Soundness

Given premisses Γ and a formula A, to show that A is *not* provable from Γ, it suffices to show that there is a valuation $\mathcal{V}$ such that $\mathcal{V}^*(A) = 0$ and for all $B \in \Gamma$, $\mathcal{V}^*(B) = 1$.

Note how powerful this method is: the claim that $\Gamma \nvdash A$ is a *universal* claim: it says that there is *no* proof of A from Γ (equivalently, that *all* proofs with premisses from Γ do not have A as a conclusion). So to check this, one would need to check all proofs. But when applying the Soundness method, we have to check an *existential* claim: we only have to provide a single counterexample in form of a valuation. This is much easier in general.

Example. One commits the fallacy of affirming the consequent when one reasons from *if p, then q* and *q* to *p*. Using Soundness, we can prove that this is indeed a fallacy. Let $\mathcal{V}$ be a valuation with $\mathcal{V}(p) = 0$ and $\mathcal{V} = 1$. Then $\mathcal{V}^*(p \to q) = 1$ and $\mathcal{V}^*(q) = 1$, but $\mathcal{V}^*(p) = 0$. Thus, p is not provable from q and $p \to q$.

Note that we now have a very useful asymmetry between syntactic and semantic consequence. To show that $\Gamma \models A$ is a universal claim over all valuations (difficult to check), but with Soundness we only need to show that $\Gamma \vdash A$ which is an existential claim over proofs (easy to check). Conversely, $\Gamma \nvdash A$ is a universal claim over all proofs (difficult), but $\Gamma \not\models A$ is an existential claim over valuations (easy). Soundness allows us in both cases to pick the easy claim and derive the difficult one.

2.11 Completeness

Note that Soundness only establishes that if we show $\Gamma \vdash A$, then it is also the case that $\Gamma \models A$, but not that whenever $\Gamma \models A$ is the case, there must be a proof of A. That is, we have not yet ruled out that we could find ourselves in a situation where $\Gamma \models A$, but $\Gamma \nvdash A$. So there may still be semantic consequences (difficult to check) that we cannot establish by a formal proof (easy to check). The *Completeness* theorem shows that this cannot be the case.

Theorem 2.25 (Completeness of the Propositional Calculus)**.** *For all sets of formulae* Γ *and formulae* A*, if* $\Gamma \models A$*, then* $\Gamma \vdash A$*.*

Before going into the proof, note that Completeness has a very different status than Soundness. In our mathematical theorising before formalising logic, we tacitly assume that our proof methods preserve truth. Without this assumption, mathematics couldn't even begin to happen. The Soundness result is merely the translation of this assumption to the formal level. We have no tacit assumption of Completeness: nothing about our mathematical practice would require us to accept that all semantic consequences of our assumptions could be proven. Thus, Completeness is a result that is a genuine meta-logical insight. Assuming that our formalisations of truth and proof are faithful to our semi-formal practice, the Completeness result allows us to infer that all semantic consequences of assumptions we make in mathematics can be demonstrated by mathematical proof.

To establish Completeness, we will show the *Satisfiability Theorem* in propositional logic.

Theorem 2.26 (Satisfiability Theorem)**.** *If Γ is a consistent set of formulae, then Γ is satisfiable. (That is: if $\Gamma \nvdash \bot$, then there is a valuation $\mathcal{V}$ such that $\mathcal{V}^*(B)$ for all $B \in \Gamma$.)*

First note why this suffices to show completeness.

Proof of Completeness from Satisfiability. Towards a *reductio* assume that there is a set Γ and formula A such that $\Gamma \models A$ and $\Gamma \nvdash A$.

Recall Theorem 2.14: $\Gamma \vdash A$ iff $\Gamma \cup \{\neg A\}$ is inconsistent.

By the contrapositive of Theorem 2.14, $\Gamma \nvdash A$ entails that $\Gamma \cup \{\neg A\}$ is consistent. By the Satisfiability Theorem, it follows that $\Gamma \cup \{\neg A\}$ is satisfiable. This means there is a valuation $\mathcal{V}$ with $\mathcal{V}^*(A) = 0$ and for all $B \in \Gamma$, $\mathcal{V}^*(B) = 1$. Thus $\Gamma \not\models A$ by definition of semantic consequence. Contradiction to our assumption that $\Gamma \models A$.

Thus there are no Γ and A with $\Gamma \models A$ and $\Gamma \nvdash A$. Hence for all Γ and A, if $\Gamma \models A$, then also $\Gamma \vdash A$. □

But to show the Satisfiability theorem we will need a little bit more Set Theory.

Definition 31. A set s is called **countable** if there is a function $f : s \to \mathbb{N}$ such that for all $x \in s$ and $y \in s$, if $x \neq y$ then $f(x) \neq f(y)$ (i.e. no two elements of s are mapped to the same number). If s is infinite and countable, it is **countably infinite**.

Clearly, all subsets of the natural numbers are countable (the even numbers, the odd numbers, the prime numbers...). But also intuitively 'larger' sets are countable.

Example. The integers $\mathbb{Z}$ (the natural numbers and their negatives) are countable. This is shown by this function:

$$f(z) = \begin{cases} 2 \cdot z, \text{ if } z > 0 \\ -2 \cdot z - 1, \text{ if } z < 0 \\ 0, \text{ if } z = 0 \end{cases}$$

All positive numbers are mapped to even numbers and all negative numbers are mapped to odd numbers, so there is no overlap.

Clearly, if a set X is countable and $Y \subseteq X$ is countable, then Y is countable. This is because if $f : X \to \mathbb{N}$ is a function showing that X is countable, then the same function restricted to Y shows that Y is countable.

The following alternative characterisation of countability is sometimes more useful.

Theorem 2.27. *A set s is countable iff $s = \emptyset$ or there is a function $g : \mathbb{N} \to s$ such that for each $x \in s$ there is a $n \in \mathbb{N}$ such that $g(n) = x$.*

Proof. Left-to-right. Let s be arbitrary and assume there is a function $f : s \to \mathbb{N}$ such that for all $x \in s$ and $y \in s$, if $x \neq y$, then $f(x) \neq f(y)$. It is to show that there is a function $g : \mathbb{N} \to s$ as in the theorem. If $s = \emptyset$ we are done; so assume s is nonempty and let $x_0 \in s$ be any member of s. Construct a function $g : \mathbb{N} \to s$ as follows: for all n such that there is an $x \in s$ with $f(x) = n$, let $g(n) = x$. For all other n, let $g(n) = x_0$. Because according to the definition of f, every $x \in s$ is mapped to a unique $n \in \mathbb{N}$, it follows that for all $x \in s$ there is a $n \in \mathbb{N}$ such that $g(n) = x$.

Right-to-left. If $s = \emptyset$, there is nothing to show. So let $s \neq \emptyset$ be arbitrary and assume there is a function $g : \mathbb{N} \to s$ as in the theorem. It is to show that there is a function $f : s \to \mathbb{N}$ such that for all $x \in s$ and $y \in s$, if $x \neq y$, then $f(x) \neq f(y)$. Define f as follows. For every $x \in s$, let n be the smallest number with $g(n) = s$ and define $f(x) = n$. Now, let $x \in s$ and $y \in s$. Show that if $f(x) = f(y)$, then $x = y$ (the contrapositive of the condition on f). If $f(x) = f(y) = n$ then $x = g(n)$ and $y = g(n)$. Because g is a function that maps every n to a unique member of s, this means that $x = y$. □

This is useful because if we have a countable set s and a function g as in the theorem, we can write s as $s = \{g(i) \mid i \in \mathbb{N}\}$ or, sloppily, as $s = \{g(0), g(1), g(2), ...\}$. We call this **enumerating** the set s. If we know there is a function g, we need not even mention it and can write an enumeration of s directly as $s = \{x_0, x_1, x_2, ...\}$ or, less sloppily, as $s = \{x_i \mid i \in \mathbb{N}\}$. Sometimes, we write even finite sets s as $s = \{x_i \mid i \in \mathbb{N}\}$ (in this case, we are simply multiply-mentioning some element as in the proof above).

We can also extend our set-theoretic operations of union and intersection to the countably infinite.

Definition 32. Let $\{s_i \mid i \in \mathbb{N}\}$ be a countable set of sets. We write $\bigcup_{i\in\mathbb{N}} s_i$ for the **union of** the s_n and define it as follows.

$\bigcup_{i\in\mathbb{N}} s_i = \{x \mid \text{there is a number } i \text{ such that } x \in s_i\}$.

Similarly, we write $\bigcap_{i\in\mathbb{N}} s_i$ for the **intersection of** the s_i and define it as follows.

$\bigcap_{i\in\mathbb{N}} s_i = \{x \mid \text{for all numbers } i \text{ it is the case that } x \in s_i\}$.

The following is a central and historically important result in Set Theory. It will be useful for proving completeness.

Theorem 2.28. *The union of countably many countable sets is countable. (More precisely: if all members of $\{s_i \mid i \in \mathbb{N}\}$ are countable, then $\bigcup_{i\in\mathbb{N}} s_i$ is countable.)*

Proof. Let $\{s_i \mid i \in \mathbb{N}\}$ be an arbitrary countable set where all members are countable.

We may make the additional assumption that they are all countably infinite so that each s_i can be written as $s_i = \{s_{i,j} \mid j \in \mathbb{N}\}$ without multiply mentioning members. (**Remark**: This uses the Axiom of Countable Choice, a generally harmless assumption.)

We may assume this because if some of the s_i are finite, we just add additional "dummy" elements. This makes the set $\bigcup_{i\in\mathbb{N}} s_i$ larger, but since obviously subsets of countable sets are countable, if we show that $\bigcup_{i\in\mathbb{N}} s_i$ is countable with extra elements, we have also shown that it is countable without extra elements.

We may also assume that all s_i are pairwise disjoint, i.e. for all $i \neq j$ there is no x such that $x \in s_i$ and $x \in s_j$. We may assume this because if there are such x, we just remove them from all sets but one and the set $\bigcup_{i\in\mathbb{N}} s_i$ will be the same.

Assumptions like the above two are sometimes called assumptions **without loss of generality** (or 'wlog' for short). This is because we make additional assumptions and, normally, additional assumptions make a theorem less general (as it then applies in fewer cases). But since we have stated how we could proceed when the assumptions are not the case and still recover the theorem, we do not lose generality by assuming them.

We need to find a function $f : \bigcup_{i\in\mathbb{N}} s_i \to \mathbb{N}$. Because we have assumed the s_i to be pairwise disjoint, this means we need to find a function f that maps each $s_{i,j}$ to a number. (If the s_i were not disjoint, it could be that there are i, j and l, k with $i \neq l$, but $s_{i,j} = s_{l,k}$, so the members of $\bigcup_{i\in\mathbb{N}} s_i \to \mathbb{N}$ would not be individuated by identifying them as some $s_{i,j}$.) The following is such a function.

$$f(s_{i,j}) = i + \frac{(i+j)(i+j+1)}{2}$$

This is the **Cantor Pairing Function**.

It looks very strange, but there is a principled way to find this function (it's not like Cantor woke up one morning with this function in his head). Here is how you would go about finding it. Consider the following strategy for mapping the $s_{i,j}$ into the numbers.

1. Map $s_{0,0}$ to 0.
2. Map $s_{0,1}$ to 1 and $s_{1,0}$ to 2.
3. Map $s_{0,2}$ to 3 and $s_{1,1}$ to 4 and $s_{2,0}$ to 5.
4. Map $s_{0,3}$ to 6 and $s_{1,2}$ to 7 and $s_{2,1}$ to 8 and $s_{3,0}$ to 9.

And so on. That is, in every step, we take one more element of the s_i we have already begun mapping to numbers and start with a new set s_i. But this is an 'and so on' that should really be made precise.

Suppose we want to find the number that $s_{i,j}$ is mapped to for some fixed i and j. We know that we start mapping the members of s_i in the $(i+1)$th step of the above construction and we get to its member $s_{i,j}$ in j more steps. Thus, we find the number that we map $s_{i,j}$ to in the $(i+j+1)$th step. Note that in the construction, in step n we make exactly n new assignments. Thus when we begin the $(i+j+1)$th step, we have found as many mappings as the sum of $(i+j)$ and all smaller numbers.

We have shown in the very beginning that the sum of $(i+j)$ and all smaller numbers is $\frac{(i+j)(i+j+1)}{2}$. This is how many numbers we have already assigned in the beginning of the $(i+j+1)$st step. Now, within that step, we will get to $s_{i,j}$ after having already assigned $j-1$ numbers in that step. Thus the number assigned to $s_{i,j}$ is the next one: $i+\frac{(i+j)(i+j+1)}{2}$. (Note that this is just a guide on how to find the Cantor Pairing Function: to make precise that the Function is the result of the 'and so on' procedure above, one would define it a recursion and then prove by induction that it delivers the Function.)

It is left to show that the Cantor Pairing Function actually does what we want. Obviously, f maps every member of $\bigcup_{i\in\mathbb{N}} s_i$ to a number. But we still need to show that for all i, j, k and l, if $s_{i,j} \neq s_{k,l}$, then $f(s_{i,j}) \neq f(s_{k,l})$. We show this by contraposition. Assume that $f(s_{i,j}) = f(s_{k,l})$ and show that $s_{i,j} = s_{k,l}$. If $f(s_{i,j}) = f(s_{k,l})$, then the following is the case by definition:

$$i+\frac{(i+j)(i+j+1)}{2} = k+\frac{(k+l)(k+l+1)}{2}.$$

First show that $i+j=k+l$. Towards a *reductio*, assume that $i+j\neq k+l$. This means either $i+j<k+l$ or $i+j>k+l$. As either case would proceed analogously, assume $i+j<k+l$. To simplify the following computation, let $n=i+j$ and $m=(k+l)-(i+j)$ (so $m=k+l-n$). Then note the following:

$$\begin{aligned}
i+\frac{(i+j)(i+j+1)}{2} &= k+\frac{(k+l)(k+l+1)}{2} \quad \text{substitute } n \text{ and } m\text{:}\\
i+\frac{n(n+1)}{2} &= k+\frac{(n+m)(n+m+1)}{2}\\
i-k &= \frac{(n+m)(n+m+1)}{2}-\frac{n(n+1)}{2}\\
&= \frac{n^2+2nm+m^2+n+m}{2}-\frac{n^2+n}{2}\\
&= \frac{2nm+m^2+m}{2} = nm+\frac{m^2+m}{2} = nm+\frac{m(m+1)}{2}
\end{aligned}$$

Now note that we assumed that $i+j<k+l$, so $m\geq 1$. This means that $nm+\frac{m(m+1)}{2}\geq n+1$. So $i-k>n$. This means that $i-k>i+j$, which means that $0>j+k$. But this cannot be, contradiction. Thus by *reductio*, $i+j=k+l$. But this entails that $i+j+1=k+l+1$, so also:

$$\frac{(i+j)(i+j+1)}{2} = \frac{(k+l)(k+l+1)}{2}.$$

But with $i+\frac{(i+j)(i+j+1)}{2} = k+\frac{(k+l)(k+l+1)}{2}$, this means that $i=k$ and since we know that $i+j=k+l$, it follows that $j=l$. This was to show. □

From this theorem, we can straightforwardly show that we only have countably many formulae.

Theorem 2.29. *The set* `wff` *is countable.*

Proof. We show that the set of words over the alphabet $\texttt{At}\cup\{\neg,\wedge,\vee,\rightarrow\}$ is countable. Show by induction that for each $n>0$, the set W_n of words of length n is countable.

- Base case. $n=1$. The alphabet is countable: this is shown by the function f that maps p, q, r, $\bot$,

$\neg$, $\wedge$, $\vee$ and $\to$ to the numbers 0–7, respectively, and all atoms p_i to $8 + i$. Since every word of length 1 corresponds to a single symbol, the same mapping shows that the set of words of length 1 is countable.

- Inductive step. Induction hypothesis: Assume for a fixed but arbitrary n that the set of words of length n is countable. Show that the set of words of length $n+1$ is countable.

  Enumerate the alphabet as $\{x_n \mid n \in \mathbb{N}\}$. For each symbol x_n, let $s_n = \{y^\frown\langle x_n\rangle \mid y \in W_n\}$. By the induction hypothesis, there is a function $f : W_n \to \mathbb{N}$ showing that W_n is countable. We can find a function $f' : s_n \to \mathbb{N}$ by defining $f'(y^\frown\langle x_n\rangle) = f(y)$. This shows that s_n is countable: if $x \in s_n$, $x' \in s_n$ and $x \neq x'$, then there are $y \in W_n$, $y' \in W_n$ such that $x = y^\frown\langle x_n\rangle$ and $x' = y'^\frown\langle x_n\rangle$. So since $x \neq x'$, $y \neq y'$. Hence $f(y) \neq f(y')$ and therefore $f'(x) \neq f'(x')$.

  Each word of length $n+1$ can be formed by appending a symbol to a word of length n, so it follows that $W_{n+1} = \bigcup_{n\in\mathbb{N}} s_n$. By the previous theorem, this set is countable.

This concludes the induction. The set of all words is $\bigcup_{n\in\mathbb{N}} W_n$, which is countable by the previous theorem. As `wff` is a subset of this set, it is countable as well. □

Our task is to prove the Satisfiability Theorem. To show that a set is satisfiable, we need to provide a valuation that makes all members true. How would one go about finding a valuation from a set? The only thing we know is that all members of the set should be true given the valuation.

At the very least, we can take all atoms from the set and assign them value 1. But this is not sufficient: a consistent set could contain a formula like $p \vee q$ without containing either p or q, so just defining a valuation by assigning 1 to all atoms would not result in a valuation that makes $p \vee q$ true. We also cannot just randomly pick one of p or q in such a case, as it could be the case that other formulae in the set rule out making p true—and possibly for non-trivial reasons (the set could contain complex formulae like $A \to \neg p$ for highly complex A).

The trick is to take a consistent set and **extend** it to a larger set that decides the truth value of all the atoms. The following definition states what the goal of such an extension is.

Definition 33. A set of formulae Γ is called **maximally consistent** iff$_{\text{Def}}$ it is consistent and all proper supersets Γ' of Γ are inconsistent.

There are many maximally consistent sets.

Example. If $\mathcal{V}$ is a valuation, then the set $\{A \mid \mathcal{V}^*(A)\}$ (the set of all formulae that are true given $\mathcal{V}$) is maximally consistent.

First show that it is consistent: if $\{A \mid \mathcal{V}^*(A) = 1\}$ is inconsistent, then $\{A \mid \mathcal{V}^*(A) = 1\} \vdash \bot$. So by Soundness, $\{A \mid \mathcal{V}^*(A) = 1\} \models \bot$. But by definition of semantic consequence, it would follow that $\mathcal{V}^*(\bot) = 1$, which cannot be. Then show that it is maximal: if $\Gamma \supset \{A \mid \mathcal{V}^*(A) = 1\}$, then there is some $A \in \Gamma$ with $\mathcal{V}^*(A) = 0$. But then $\mathcal{V}^*(\neg A) = 1$, so $\neg A \in \{A \mid \mathcal{V}^*(A) = 1\}$. Then $\Gamma \vdash \bot$ by ($\neg$E.).

The crucial result is now that if we have a consistent set, we can always extend it to a maximally consistent one.

Theorem 2.30. *If Γ is consistent, there is a set of formulae $\hat{\Gamma} \supseteq \Gamma$ that is maximally consistent.*

Proof. Let Γ be an arbitrary consistent set. We have shown above that `wff` is countable, so enumerate it as $\texttt{wff} = \{A_i \mid i \in \mathbb{N}\}$.

By recursion on the natural numbers, now define sets Γ_n for each natural number n.

- Base case, $n = 0$: $\Gamma_0 = \Gamma$.
- Inductive step. Suppose we have already constructed Γ_n for a fixed but arbitrary n. Then define Γ_{n+1} as follows. If $\Gamma_n \cup \{A_n\}$ is consistent, let $\Gamma_{n+1} = \Gamma_n \cup \{A_n\}$, otherwise let $\Gamma_{n+1} = \Gamma_n$.

Then let $\hat{\Gamma} = \bigcup_{n \in \mathbb{N}} \Gamma_n$, i.e. for all formulae A, $A \in \hat{\Gamma}$ iff there is a number n such that $A \in \Gamma_n$. It is easy to see that if $i < j$, then $\Gamma_i \subseteq \Gamma_j$ and that all Γ_i are consistent. In particular, $\Gamma \subseteq \hat{\Gamma}$. Now show that $\hat{\Gamma}$ is maximally consistent.

First show that it is consistent. Assume towards a *reductio* that $\hat{\Gamma} \vdash \bot$. There can only be finitely many premisses $B_0, ..., B_n$ involved in this proof. For any i, if $B_i \in \hat{\Gamma}$, this means there is some n such that $B_i \in \Gamma_n$. For each B_i let n_i be the smallest number so that $B_i \in \Gamma_{n_i}$. Let $\hat{n}$ be the maximum of the n_i. This means that for all i, $B_i \in \Gamma_{\hat{n}}$. But this means that $\Gamma_{\hat{n}} \vdash \bot$. But all Γ_n are consistent, hence contradiction. Thus $\hat{\Gamma}$ is consistent.

Then show that $\hat{\Gamma}$ is maximally consistent. Assume towards a *reductio* that it is not, i.e. there is some formula $A \notin \hat{\Gamma}$ such that $\hat{\Gamma} \cup \{A\}$ is consistent. There is some number n such that $A = A_n$. Because $A \notin \hat{\Gamma}$, A_n was not added in the $(n+1)$th step of the above recursion. So $\Gamma_n \cup \{A_n\}$ is inconsistent. But $\Gamma_n \cup \{A_n\} \subseteq \hat{\Gamma} \cup \{A_n\}$. So all premisses required for the proof of $\bot$ from $\Gamma_n \cup \{A_n\}$ are in $\hat{\Gamma} \cup \{A_n\}$. Thus $\hat{\Gamma} \cup \{A_n\}$ is inconsistent. Contradiction to our assumption. Thus $\hat{\Gamma}$ is maximally consistent. □

Maximally consistent sets have a very important property: they are *deductively closed.*

Theorem 2.31. *Let $\hat{\Gamma}$ be a maximally consistent set. Then $\hat{\Gamma}$ is **deductively closed**, i.e. for all A, if $\hat{\Gamma} \vdash A$, then $A \in \hat{\Gamma}$.*

Proof. Let $\hat{\Gamma}$ be arbitrary and maximally consistent. Let A be an arbitrary formula such that $\hat{\Gamma} \vdash A$. Towards a *reductio*, assume that $A \notin \hat{\Gamma}$. Because $\hat{\Gamma}$ is maximally consistent, this means that $\hat{\Gamma} \cup \{A\}$ is inconsistent. By double-negation elimination, this means that $\hat{\Gamma} \cup \{\neg\neg A\}$ is inconsistent. Hence by Theorem 2.14, $\hat{\Gamma} \vdash \neg A$. But then because $\hat{\Gamma} \vdash A$, $\hat{\Gamma}$ is inconsistent. Contradiction to the assumption that $\hat{\Gamma}$ is maximally consistent. Thus, by *reductio*, $A \in \hat{\Gamma}$. □

Now we are ready to prove the Satisfiability Theorem and, with it, Completeness.

Proof (Satisfiability Theorem). Let Γ be an arbitrary, consistent set of formulae. Let $\hat{\Gamma} \supseteq \Gamma$ be maximally consistent. Let $\mathcal{V}$ be a valuation defined as follows: if a is an atom, $\mathcal{V}(a) = 1$ iff$_{\text{Def}}$ $a \in \hat{\Gamma}$.

Now show by induction over the construction of formulae that for all wffs A: $A \in \hat{\Gamma}$ iff $\mathcal{V}^*(A) = 1$.

- Base case. Suppose A is atomic. If $A \in \hat{\Gamma}$ then by definition $\mathcal{V}(A) = 1$, so $\mathcal{V}^*(A) = 1$. Conversely, if $\mathcal{V}^*(A) = 1$, then $\mathcal{V}(A) = 1$, which by definition means that $A \in \hat{\Gamma}$.
- Inductive steps.
  - Suppose $A = B \wedge C$ for fixed but arbitrary B and C. Induction hypothesis: $B \in \hat{\Gamma}$ iff $\mathcal{V}^*(B) = 1$ and $C \in \hat{\Gamma}$ iff $\mathcal{V}^*(C) = 1$. It is to show that $A \in \hat{\Gamma}$ iff $\mathcal{V}^*(A) = 1$.

    Left-to-right. Suppose that $A \in \hat{\Gamma}$ and show $\mathcal{V}^*(A) = 1$. By ($\wedge$E.), $\hat{\Gamma} \vdash B$ and $\hat{\Gamma} \vdash C$. By the previous theorem, $B \in \hat{\Gamma}$ and $C \in \hat{\Gamma}$ so by the IH, $\mathcal{V}^*(B) = 1$ and $\mathcal{V}^*(C) = 1$. By the Truth Table for $\wedge$, $\mathcal{V}^*(A) = 1$.

    Right-to-left. Suppose that $\mathcal{V}^*(A) = 1$ and show that $A \in \hat{\Gamma}$. By the Truth Table for $\wedge$, $\mathcal{V}^*(A) = 1$ entails that $\mathcal{V}^*(B) = 1$ and $\mathcal{V}^*(C) = 1$. By the IH, $B \in \hat{\Gamma}$ and $C \in \hat{\Gamma}$, so by ($\wedge$I.) $\hat{\Gamma} \vdash A$ and so by the previous theorem $A \in \hat{\Gamma}$.
  - Suppose $A = \neg B$ for a fixed but arbitrary B. Induction hypothesis: $B \in \hat{\Gamma}$ iff $\mathcal{V}^*(B) = 1$.

    Left-to-right. Suppose that $A \in \hat{\Gamma}$ and show $\mathcal{V}^*(A) = 1$. Assume towards a *reductio* that $\mathcal{V}^*(A) = 0$. By the Truth Table for $\neg$, $\mathcal{V}^*(B) = 1$, so by the IH $B \in \hat{\Gamma}$. But by assumption $A \in \hat{\Gamma}$, i.e. $\neg B \in \hat{\Gamma}$. So $\hat{\Gamma}$ is inconsistent. Contradiction to $\hat{\Gamma}$ being maximally consistent. Hence by *reductio*, $\mathcal{V}^*(A) = 1$.

    Right-to-left. Suppose that $\mathcal{V}^*(A) = 1$ and show that $A \in \hat{\Gamma}$. By the Truth Table for negation, $\mathcal{V}^*(B) = 0$, so by the IH $B \notin \hat{\Gamma}$. Because $\hat{\Gamma}$ is maximally consistent, $\hat{\Gamma} \cup \{B\}$ is inconsistent, i.e. $\hat{\Gamma} \cup \{B\} \vdash \bot$. So by ($\neg$I.), $\hat{\Gamma} \vdash \neg B$. By the previous theorem, $\neg B \in \hat{\Gamma}$, so $A \in \hat{\Gamma}$.
  - Suppose $A = B \vee C$ for fixed but arbitrary B and C. Induction hypothesis: $B \in \hat{\Gamma}$ iff $\mathcal{V}^*(B) = 1$ and $C \in \hat{\Gamma}$ iff $\mathcal{V}^*(C) = 1$. It is to show that $A \in \hat{\Gamma}$ iff $\mathcal{V}^*(A) = 1$.

    Left-to-right. Suppose that $A \in \hat{\Gamma}$ and show $\mathcal{V}^*(A) = 1$. Towards a *reductio*, assume that $\mathcal{V}^*(A) = 0$. By the Truth Table for $\vee$, $\mathcal{V}^*(B) = 0$ and $\mathcal{V}^*(C) = 0$. By the IH, this means that $B \notin \hat{\Gamma}$ and $C \notin \hat{\Gamma}$. As $\hat{\Gamma}$ is maximally consistent, this means that $\hat{\Gamma} \cup \{B\}$ and $\hat{\Gamma} \cup \{C\}$ are inconsistent, i.e. $\hat{\Gamma} \cup \{B\} \vdash \bot$ and $\hat{\Gamma} \cup \{C\} \vdash \bot$. But then because $A \in \hat{\Gamma}$, by ($\vee$E.) it follows that $\hat{\Gamma} \vdash \bot$. Contradiction. Hence by *reductio*, $\mathcal{V}^*(A) = 1$.

    Right-to-left. Suppose that $\mathcal{V}^*(A) = 1$ and show that $A \in \hat{\Gamma}$. By the Truth Table for $\vee$, $\mathcal{V}^*(A) = 1$ means there are two cases.

    * Case 1. $\mathcal{V}^*(B) = 1$. By the IH, $B \in \hat{\Gamma}$, so by ($\vee$I.) $\hat{\Gamma} \vdash A$. By the previous theorem, $A \in \hat{\Gamma}$.
    * Case 2. $\mathcal{V}^*(C) = 1$. By the IH, $C \in \hat{\Gamma}$, so by ($\vee$I.) $\hat{\Gamma} \vdash A$. By the previous theorem, $A \in \hat{\Gamma}$.

    By the IH, $B \in \hat{\Gamma}$ and $C \in \hat{\Gamma}$, so by ($\wedge$I.) $\hat{\Gamma} \vdash A$ and so by the previous theorem $A \in \hat{\Gamma}$.
  - Suppose $A = B \to C$ for fixed but arbitrary B and C. Induction hypothesis: $B \in \hat{\Gamma}$ iff $\mathcal{V}^*(B) = 1$ and $C \in \hat{\Gamma}$ iff $\mathcal{V}^*(C) = 1$. It is to show that $A \in \hat{\Gamma}$ iff $\mathcal{V}^*(A) = 1$.

Left-to-right. Suppose that $A \in \hat{\Gamma}$ and show $\mathcal{V}^*(A) = 1$. Assume towards a *reductio* that $\mathcal{V}^*(A) = 0$. By the Truth Table for →, this means that $\mathcal{V}^*(B) = 1$ and $\mathcal{V}^*(C) = 0$. By the induction hypothesis, this means that $B \in \hat{\Gamma}$ and $C \notin \hat{\Gamma}$. Because $A \in \hat{\Gamma}$ and $B \in \hat{\Gamma}$, $\hat{\Gamma} \vdash C$ by (→E.). So $C \in \hat{\Gamma}$ by the previous theorem. Contradiction. Hence by *reductio*, $\mathcal{V}^*(A) = 1$.

Right-to-left. Suppose that $\mathcal{V}^*(A) = 1$ and show that $A \in \hat{\Gamma}$. As $\mathcal{V}^*(A) = 1$, there are two cases: $\mathcal{V}^*(B) = 0$ and $\mathcal{V}^*(C) = 1$ (because $B \to C \approx \neg B \vee C$).

* Case 1. $\mathcal{V}^*(B) = 0$. By the IH, $B \notin \hat{\Gamma}$, so $\hat{\Gamma} \cup \{B\}$ is inconsistent, i.e. $\hat{\Gamma} \cup \{B\} \vdash \bot$. By (RAA) with an empty discharge, $\hat{\Gamma} \cup \{B\} \vdash C$, so by (→I.), $\hat{\Gamma} \vdash B \to C$. By the previous theorem, $A \in \hat{\Gamma}$.
* Case 2. $\mathcal{V}^*(C) = 1$. By the IH, $C \in \hat{\Gamma}$, so by (→I.) with an empty discharge, $\hat{\Gamma} \vdash B \to C$. By the previous theorem, $A \in \hat{\Gamma}$.

This concludes the induction. Thus for all $A \in \hat{\Gamma}$, $\mathcal{V}^*(A) = 1$. Hence since $\Gamma \subseteq \hat{\Gamma}$, for all $A \in \Gamma$, $\mathcal{V}^*(A) = 1$. Thus Γ is satisfiable. □

Note that although the Soundness theorem rested (in a sense) on our meta-language proof-methods being sound for meta-language truth, the Completeness theorem does not in any sense rest on the completeness of the meta-language.

An immediate upshot is that we can finally prove the compactness of semantic consequence.

Theorem 2.32. *For all sets of formulae Γ and formulae A: If $\Gamma \models A$ there is a finite $\Gamma' \subseteq \Gamma$ such that $\Gamma' \models A$.*

Proof. Let A and Γ be arbitrary and assume $\Gamma \models A$. By Completeness, $\Gamma \vdash A$. By compactness of $\vdash$, this means that there is a finite $\Gamma' \subseteq \Gamma$ with $\Gamma' \vdash A$. By Soundness, $\Gamma' \models A$. □

3 Predicate Logic

In propositional logic, we abstracted away from the particular contents of sentences and merely associated a truth value with each sentence. We now eliminate this abstraction and look at what kinds of contents sentences can have.

3.1 Languages and Structures

We now want to speak about particular sentences and their truth values. Amongst the sentences in our language that are useful for these purposes, the obvious candidates are the *declarative* sentences. In their simplest form, declarative sentences consist of a *noun phrase*, denoting some object, and a *verb phrase*, denoting some property. Such a sentence is true if the denotation of the noun phrase has the property denoted by the verb phrase. Some examples:

- Hypatia is mortal. (true iff the person *Hypatia* has the property *being mortal*)
- The square root of 2 is irrational. (true iff the number $\sqrt{2}$ has the property *being irrational*)

We can use *functions* to pick out the denotation of the noun phrase, as in the following examples.

- Hypatia's mother is human. (true iff the person that is the *mother-of* the person *Hypatia* has the property *being human*).
- 5 plus 6 is prime. (true iff the number that is the *sum-of* the numbers 5 and 6 has the property *being prime*).

Sometimes, properties themselves come with reference to an object.

- Hesperus is Phosphorus. (true iff the thing *Hesperus* has the property *being (equal to) Phosphorus.*)
- London is west of Amsterdam. (true iff the thing *London* has the property *being west of Amsterdam.*)
- 5 is less than 7. (true iff the number 5 has the property *being less than the number 7.*)

In these cases, it might be easier to not speak of properties, but of *relations*.

- Hesperus is Phosphorus. (true iff the things *Hesperus* and *Phosphorus* stand in the *equal* relation.)
- London is west of Amsterdam. (true iff the things *London* and *A'dam* stand in the *west-of* relation.)
- 5 is less than 7. (true iff the numbers 5 and 7 stand in the *less-than* relation.)

An useful thing about this manner of speaking is that we can speak of relations that relate more than just two objects.

- 5 is in between 3 and 7. (true iff the numbers 5, 3 and 7 stand in the *in-between* relation.)

There is much (*much*) more to say about the syntax and truth-conditions of declarative sentences. For the purposes of predicate logic, however, we are happy with the language of *names* (Hypatia, Hesperus, London), *functions* (mother-of) and *relations* (being-mortal, west-of, in-between).

Definition 34 (Language)**.** A **language** is a quadruple $\mathcal{L} = \langle \mathcal{C}, \mathcal{F}, \mathcal{R}, \alpha \rangle$ where $\mathcal{C}$, $\mathcal{F}$, $\mathcal{R}$ are pairwise disjoint sets of symbols (i.e. no symbol occurs in both $\mathcal{C}$ and $\mathcal{F}$, both $\mathcal{C}$ and $\mathcal{R}$ or both $\mathcal{F}$ and $\mathcal{R}$) and $\alpha : \mathcal{F} \cup \mathcal{R} \to \mathbb{N}$ is a function.

The members of $\mathcal{C}$ are called **constant symbols** (or names), the members of $\mathcal{F}$ are called **function symbols** and the members of $\mathcal{R}$ are called **relation symbols** (or predicate symbols).

The function α associates each function symbol and relation symbol with a natural number, the **arity** of the function/relation symbol. The arity of a function or relation symbol is how many arguments it requires. For example, in the above examples, ''s mother' is a function with arity 1 (it takes one argument), 'plus' is a function with arity 2 (it takes two arguments), 'mortal' is a relation with arity 1 and 'in between' is a relation symbol with arity 3. Note that in principle we can consider 0-ary function symbols and relation symbols, but these are usually not very interesting.

Example. Here are some examples of languages.

- The **language of arithmetic** is $\mathcal{C} = \{\text{'}0\text{'}\}$, $\mathcal{F} = \{\text{'}s\text{'}, \text{'}+\text{'}, \text{'}\cdot\text{'}\}$, $\mathcal{R} = \emptyset$ with $\alpha(\text{'}s\text{'}) = 1$, $\alpha(\text{'}+\text{'}) = 2$ and $\alpha(\text{'}\cdot\text{'}) = 2$.
- The **language of set theory** is $\mathcal{C} = \emptyset$, $\mathcal{F} = \emptyset$, $\mathcal{R} = \{\text{'}\in'\}$ with $\alpha(\text{'}\in\text{'}) = 2$.

- The **language of cardinal geography** could be something like: $\mathcal{C}$ contains a name for each place on earth, $\mathcal{F} = \emptyset$ and $\mathcal{R} = \{$'north-of', 'east-of', 'south-of', 'west-of'$\}$ with $\alpha(R) = 2$ for all $R \in \mathcal{R}$.

Note that again this is **just syntax**. We associate a set of symbols (that could be anything) with an arity to determine how a **well-formed sentence** has to look (how many arguments need to be supplied; we will define this formally shortly). But this in itself does not **constitute meaning in any way**. Nothing so far guarantees that, say, the symbol '+' denotes the function for addition.

Languages receive meanings from being *interpreted.*

Definition 35 (Structure and Interpretation)**.** Given a language $\mathcal{L} = \langle \mathcal{C}, \mathcal{F}, \mathcal{R}, \alpha \rangle$, an **$\mathcal{L}$-structure** is an ordered pair $\mathcal{M} = \langle M, I \rangle$ where:

1. M is a nonempty set (the **domain** of $\mathcal{M}$).
2. I is an **interpretation** of the symbols in $\mathcal{L}$ i.e.:
   (a) for each $c \in \mathcal{C}$, $I(c)$ is an element of M.
   (b) for each $F \in \mathcal{F}$, $I(F)$ is a function that maps $\alpha(F)$-many elements of M to an element of M.
   (c) for each $R \in \mathcal{R}$, $I(R)$ is a function that maps $\alpha(R)$-many elements of M to a truth value.

A little bit of notation: given a set X and a number n we write X^n for the set of all n-tuples with elements from X. Then we can write the definition of interpretation very compactly:

(a) for each $c \in \mathcal{C}$, $I(c) \in M$.
(b) for each $F \in \mathcal{F}$, $I(F) : M^{\alpha(f)} \to M$.
(c) for each $R \in \mathcal{R}$, $I(R) : M^{\alpha(r)} \to \{0, 1\}$.

For example, we can give the **intended interpretation** of the language of arithmetic by the following structure:

- $M = \mathbb{N}$
- $I(`0') = 0$,
- $I(`s')$ is the function $S : \mathbb{N} \to \mathbb{N}$ such that for all $i \in \mathbb{N}$, $S(i) = i + 1$.
- $I(` + ')$ is the function $p : \mathbb{N}^2 \to \mathbb{N}$ such that for all $i, j \in \mathbb{N}$, $p(i, j) = i + j$.
- $I(` \cdot ')$ is the function $m : \mathbb{N}^2 \to \mathbb{N}$ such that for all $i, j \in \mathbb{N}$, $m(i, j) = i \cdot j$.

We will not always keep around the quotation marks and write something like: for all numbers i, j define $I(+)(i, j) = i + j$. Here we use our *meta-language operation* $+$ (which means addition) to define an interpretation of *symbol* $+$ (which in itself has no meaning). As the symbol occurs in an interpretation function, we should not be confused about this. (But we should still be careful.)

Of course, there are many other (unintended?) interpretations of a language. We could just as well interpret the language of arithmetic as follows.

- $M = \{\heartsuit, \clubsuit\}$

- $I(‘0’) = \heartsuit$,
- $I(‘s’)$ is the function S such that $S(\heartsuit) = \heartsuit$ and $S(\clubsuit) = \heartsuit$.
- $I(‘+’)$ is the function p such that $S(\heartsuit, \clubsuit) = \clubsuit$ and $S(\clubsuit, \heartsuit) = \heartsuit$.
- $I(‘\cdot’)$ is the function m such that $S(\heartsuit, \clubsuit) = \heartsuit$ and $S(\clubsuit, \heartsuit) = \heartsuit$.

Now note that, at least intuitively, we can chain together function symbols. For example, in the language of arithmetic, we should be able to talk about *the successor of the successor of the successor of 0* (or: $sss0$). Such complex expressions are called **terms**.

3.2 Terms and their Interpretation

Given a language, we can define the set of terms over that language by recursion.

Definition 36 (Terms over a Language)**.** Let $\mathcal{L} = \langle \mathcal{C}, \mathcal{F}, \mathcal{R}, \alpha \rangle$ be a language. Let $\mathcal{V} = \{x_i \mid i \in \mathbb{N}\}$ be a set of **variable symbols** (of symbols that are not part of the language). The **set of terms** $\mathtt{Tm}(\mathcal{L})$ **over** $\mathcal{L}$ is defined by the following recursion.

i. For all constant symbols $c \in \mathcal{C}$, $\langle c \rangle \in \mathtt{Tm}(\mathcal{L})$.
ii. For all variable symbols $x \in \mathcal{V}$, $\langle v \rangle \in \mathtt{Tm}(\mathcal{L})$.
iii. If $F \in \mathcal{F}$ with $\alpha(F) = n$ and t is a concatenation of n members of $\mathtt{Tm}(\mathcal{L})$, then $\langle f \rangle^\frown t \in \mathtt{Tm}(\mathcal{L})$. (sloppily: If $F \in \mathcal{F}$ with $\alpha(F) = n$ and $t_0, ..., t_{n-1} \in \mathtt{Tm}(\mathcal{L})$, then $\langle F \rangle^\frown t_0^\frown ...^\frown t_{n-1} \in \mathtt{Tm}(\mathcal{L})$.)
iv. Nothing else is a term.

We will usually not be this precise and instead of, for example, $\langle +, s, s, 0, s, 0 \rangle$ just write $+ss0s0$.

Theorem 3.1 (Unique Construction of Terms)**.** *Let $\mathcal{L} = \langle \mathcal{C}, \mathcal{F}, \mathcal{R}, \alpha \rangle$ be a language. For each $t \in \mathtt{Tm}(\mathcal{L})$, exactly one of the following is the case.*

1. *There is a $c \in \mathcal{C}$ such that $t = \langle c \rangle$.*
2. *There is a $v \in \mathcal{V}$ such that $t = \langle v \rangle$.*
3. *There are unique terms $t_0, ..., t_{\alpha(f)-1}$ and a unique $F \in \mathcal{F}$ such that $t = \langle F \rangle^\frown t_0{}^\frown ...^\frown t_{n-1}$.*

   (Not sloppy: there is a unique $F \in \mathcal{F}$ and for each $i < \alpha(F)$, there is a unique term t_i such that $t = \langle F \rangle^\frown \hat{t}$, where $\hat{t}$ is defined by: $\hat{t}_0 = t_0$; for $i > 0$, $i < n$, $\hat{t}_i = \hat{t}_{i-1}{}^\frown t_i$; $\hat{t} = \hat{t}_{n-1}$.)

Proof. The method for the proof should be familiar from the proof that the wffs of propositional logic have unique constructions. First show that no term has an initial segment that is also a term.

For *reductio* assume that there is a term t with a proper initial segment that is a term. Assume that t has minimal length with that property. It is clear that t cannot have length 1: any proper initial segment must have a shorter length and hence have length 0. But there are no terms of length 0. Thus t must be formed by (iii), i.e. there is a function symbol F with arity n and terms $t_0, ..., t_{n-1}$ such that $t = \langle F \rangle^\frown t_0^\frown ...^\frown t_{n-1}$.

Let u be a proper initial segment of t that is a term. The first element of u is the function symbol F. We know that F has arity n, so u must have the form $u = \langle F\rangle^\frown u_0^\frown ...^\frown u_{n-1}$ for terms $u_0, ..., u_{n-1}$. Since u is shorter than t, it cannot be the case that for all $i < n$, t_i and u_i have the same length. So let i be the smallest number such that u_i has a different length than t_i.

Note that because u is an initial segment of t, for all $j < i$, $t_j = u_j$ and therefore u_i and t_i begin at the same symbol in t. Again because u is an initial segment of t, this means that u_i is a proper initial segment of t_i or t_i is a proper initial segment of u_i. But clearly both t_i and u_i are shorter than t, so this contradicts our assumption that t was shortest with the property of having a proper initial segment that is a term. Contradiction. Hence all proper initial segments of all terms are not themselves terms.

Now prove the theorem. Clearly, every term t can be constructed as (1) or (2) or (3). It is to show that it can be constructed in *exactly* one of these ways. So let t be an arbitrary term and distinguish the following cases.

Case 1. There is a $c \in \mathcal{C}$ such that $t = \langle c\rangle$. Because $\mathcal{V}$ and $\mathcal{C}$ share no members, $t \neq \langle v\rangle$ for all $v \in \mathcal{V}$. Because $\mathcal{F}$ and $\mathcal{C}$ share no members, $t \neq \langle F\rangle$ for all $F \in \mathcal{F}$.

Case 2. There is a $v \in \mathcal{V}$ such that $t = \langle v\rangle$. As in Case 1, because $\mathcal{V}$, $\mathcal{C}$ and $\mathcal{F}$ share no members, $t \neq \langle c\rangle$ for all $c \in \mathcal{C}$ and $t \neq \langle F\rangle$ for all $F \in \mathcal{F}$.

Case 3. There is a $F \in \mathcal{F}$ with arity n and terms $t_0, ..., t_{n-1}$ such that $t = \langle F\rangle^\frown t_0^\frown ...^\frown t_{n-1}$. We cannot be in case (1) or (2) because $\mathcal{F}$, $\mathcal{C}$ and $\mathcal{V}$ share no members. Assume towards a *reductio* that there is a function symbol $G \in \mathcal{F}$ with arity m and terms $u_0, .., u_{m-1}$ such that $t = \langle G\rangle^\frown u_0^\frown ...^\frown u_{m-1}$ but $F \neq G$ or there is some i such that $u_i \neq t_i$. Because the first symbol in t is F it follows that $G = F$ and hence also $m = n$. Thus there must be some i with $u_i \neq t_i$. Let i be minimal with that property. Then either u_i is a proper initial segment of t_i or t_i is a proper initial segment of u_i. Neither can be the case. Contradiction. Hence by *reductio*, there is no such G and terms $u_0, ..., u_{m-1}$. This was to show.

This concludes the proof. □

With unique construction in place we can again allow ourselves some notational conventions to make terms easier to read. Instead of $+ss0s0$ we may also write $+(ss0, s0)$ (i.e. for function symbols with arity > 1, we add parentheses and commas for readability). For *binary* function symbols like $+$ we may also sometimes use the familiar notation $ss0 + s0$. When we do so, we have to be careful to add parentheses for disambiguation.

Now, we can define what it means to assign them an interpretation under a structure. Intuitively, a term should be interpreted to a member of the domain of a structure (i.e. terms are just complex names for things).

Definition 37 (Interpretation of Terms)**.** Let $\mathcal{L} = \langle \mathcal{C}, \mathcal{F}, \mathcal{R}, \alpha\rangle$ be a language, $\mathcal{M} = \langle M, I\rangle$ be an $\mathcal{L}$-structure and $f : \mathcal{V} \to M$ a function that assigns to each variable a member of the domain of $\mathcal{M}$ (we call such f **assignment functions**). For each $t \in \mathtt{Tm}(\mathcal{L})$ define $I^f(t)$, its **interpretation of t in $\mathcal{M}$ under f** by recursion.

- Base case 1. $t = c$ for some $c \in \mathcal{C}$. Then $I^f(t) =_{\text{Def}} I(c)$.
- Base case 2. $t = x$ for some $x \in \mathcal{V}$. Then $I^f(t) =_{\text{Def}} f(x)$.
- Recursive step. $t = Ft_0 ... t_{n-1}$ for a function symbol F with arity n and terms t_0 ... t_{n-1}. Then $I^f(t) =_{\text{Def}} I(F)(I^f(t_0), ..., I^f(t_{n-1}))$.

For the recursive step, recall that $I(F)$ is a function from M^n to M and that for each $i < n$, $I^f(t_i)$ is a member of M. Thus in the recursive step $I^f(t)$ is the result of applying the function $I(F)$ to the interpretations of the terms t_i.

As an example, recall the intended interpretation I of the language of arithmetic and the term $sss0$. Let f be any assignment function (it does not matter here as this term contains no variables). We can compute $I^f(sss0)$ as follows:

- $sss0$ is as in the recursive step: s is a unary function symbol and $ss0$ is a term.
  So $I^f(sss0) = I(s)(I^f(ss0)) = I^f(ss0) + 1$.
- $ss0$ is as in the recursive step: s is a unary function symbol and $s0$ is a term.
  So $I^f(ss0) = I(s)(I^f(s0)) = I^f(s0) + 1$.
- $s0$ is as in the recursive step: s is a unary function symbol and 0 is a term.
  So $I^f(s0) = I(s)(I^f(0)) = I^f(0) + 1$.
- 0 is as in a base case. $I^f(0) = I(0) = 0$.
- Now ascend back through the recursion: $I^f(s0) = I^f(0)+1 = 0+1 = 1$, so $I^f(ss0) = I^f(s0)+1 = 1+1$, so $I^f(sss0) = I^f(ss0) + 1 = 1 + 1 + 1 = 3$.

Thus the interpretation of $sss0$ in I is the number 3.

3.3 The Language of Predicate Logic

We are yet missing one important part of our language. Instead of just talking about the properties of definite things, we may also want to claim that *all* things or *some* things have a property.

- Everything is extended in space.
- All humans are mortal.
- Some numbers are prime.

To be able to express such things, we extend our vocabulary with the *quantifiers* $\forall$ ('all') and $\exists$ ('exists', 'there is'). We can now define the well-formed formulae of predicate logic over a language. (Note that the definition already uses our notational conventions about not explicitly mentioning quotation and $\langle\ \rangle$.)

Definition 38. Let $\mathcal{L}$ be a language. Define by recursion the set $\mathtt{wff}(\mathcal{L})$.

i. If t_0 and t_1 are terms over $\mathcal{L}$, then $t_0 \equiv t_1 \in \mathtt{wff}(\mathcal{L})$.
ii. If R is an n-ary relation symbol of $\mathcal{L}$ and $t_0, ..., t_{n-1}$ are terms over $\mathcal{L}$, then $Rt_0, ..., t_{n-1} \in \mathtt{wff}(\mathcal{L})$.

iii. $\bot \in \texttt{wff}(\mathcal{L})$.

iv. If $A \in \texttt{wff}(\mathcal{L})$, then also $\neg A \in \texttt{wff}(\mathcal{L})$.

v. If $A \in \texttt{wff}(\mathcal{L})$ and $B \in \texttt{wff}(\mathcal{L})$, then also $(A \wedge B) \in \texttt{wff}(\mathcal{L})$, $(A \vee B) \in \texttt{wff}(\mathcal{L})$ and $(A \to B) \in \texttt{wff}(\mathcal{L})$.

vi. If $A \in \texttt{wff}(\mathcal{L})$ and $x \in \mathcal{V}$, then also $\forall x A \in \texttt{wff}(\mathcal{L})$ and $\exists x A \in \texttt{wff}(\mathcal{L})$

vii. Nothing else is a member of $\texttt{wff}(\mathcal{L})$.

We will call the formulae formed by (i), (ii) and (iii) the **atomic** formulae of predicate logic (over some language). The symbol '$\equiv$' is called *identity*. We use $\equiv$ to distinguish the object-language symbol for equality from our meta-language use of $=$. It is straightforward to prove a Unique Construction theorem and we omit the proof here. Again, we may use our usual notational conventions for the propositional logic connectives (e.g. dropping outer parentheses) and for a formula as in (ii) we may add commas and parentheses like $R(t_0, ..., t_{n-1})$. For *binary* relation symbols we may also write $t_0 R t_1$ (and sometimes add parentheses around this for disambiguation).

We will however **never** drop outer parentheses for formulae directly occurring under a quantifier, e.g. never write $\forall x A \wedge B$ or $\exists x A \wedge B$ instead of $\forall x(A \wedge B)$ or $\exists x(A \wedge B)$. This is because doing so would introduce an ambiguity regarding the following concept. If a variable x occurs in a formula A, then in the formulae $\forall x A$ (or $\exists x A$) we say that x is **bound** by the quantifier $\forall$ (or $\exists$). A variable that is not bound is **free**. We can define the set of free variables occurring in a formula A by recursion.

Definition 39. Let $\mathcal{L}$ be a language. For each $t \in \texttt{wff}(\mathcal{L})$, define the set $\texttt{var}(t)$ of **variables** in t as all variables occurring in t.

Definition 40. Let $\mathcal{L}$ be a language. For each $A \in \texttt{wff}(\mathcal{L})$, define the set $\texttt{frvar}(A)$ of **free variables** in A as follows.

i. If t_0 and t_1 are terms over $\mathcal{L}$, then $\texttt{frvar}(t_0 \equiv t_1) = \texttt{var}(t_0) \cup \texttt{var}(t_1)$.

ii. If R is an n-ary relation symbol and $t_0, ..., t_{n-1}$ are terms, then $\texttt{frvar}(R t_0, ..., t_{n-1}) = \bigcup_{i<n} \texttt{var}(t_i)$.

iii. $\texttt{frvar}(\bot) = \emptyset$.

iv. If $A = \neg B$, then $\texttt{frvar}(A) = \texttt{frvar}(B)$.

v. If $A = B \wedge C$ or $A = B \vee C$ or $A = B \to C$, then $\texttt{frvar}(A) = \texttt{frvar}(B) \cup \texttt{frvar}(C)$.

vi. If $A = \forall x B$ or $A = \exists x B$, then $\texttt{frvar}(A)$ is the set $\texttt{frvar}(B)$ without x.

Some examples are as follows. Consider the following formulae from the language of arithmetic.

1. In $\exists x \cdot xx \equiv y$, the variable x is bound, but the variable y is free.

2. In $\forall x \exists y + yy \equiv x$ both x and y are bound and no variable is free.

3. In $(+xy \equiv z \wedge \exists x + xx \equiv x)$ all of x, y and z are free. There is a **subformula** in which the variable x is bound, but in the **whole formula**, x is free.

Indeed keeping track of what is bound by a quantifier is so important that we will almost always write parentheses around formulae under quantifiers, so we can see what they bind at a glance. That is, we might write the above examples like so.

1. $\exists x(\cdot xx \equiv y)$.
2. $\forall x \exists y(+yy \equiv x)$.
3. $(+xy \equiv z \wedge \exists x(+xx \equiv x))$.

Such parentheses are not strictly speaking necessary (i.e. we can prove Unique Construction without them), but they help *us* to read the formulae. Using our other notational conventions, we indeed may write these same formulae as follows.

1. $\exists x(x \cdot x \equiv y)$.
2. $\forall x \exists y(y + y \equiv x)$.
3. $x + y \equiv z \wedge \exists x(x + x \equiv x)$.

It is often important to know which variables occur free in a formula. We will sometimes talk about a formula A with free variables $x_0, ..., x_{n-1}$ by writing $A(x_0, ...x_{n-1})$ to indicate this.

3.4 Satisfaction and Models

When we have a language and a structure for that language and also have an assignment function for the variables, we have everything we need to determine when a formula is **true in a structure given an assignment**. That is, a structure with an assignment is in predicate logic what a valuation was in propositional logic. When a formula A is true in a structure $\mathcal{M}$ and assignment f, we say that $\langle \mathcal{M}, f \rangle$ is a **model of** A or $\langle \mathcal{M}, f \rangle$ **satisfies** A. We write this as $\mathcal{M}, f \models A$ (read the $\models$ as "models").

Definition 41. Let $\mathcal{L}$ be a language and $\mathcal{M} = \langle M, I \rangle$ be an $\mathcal{L}$-structure. Let $f : \mathcal{V} \to M$ be an assignment function. Then define by recursion on the construction of the formulae:

i. If t_0 and t_1 are terms over $\mathcal{L}$, then $\mathcal{M}, f \models (t_0 \equiv t_1)$ iff$_{\text{Def}}$ $I^f(t_0) = I^f(t_1)$.

ii. If R is an n-ary relation symbol and $t_0, ..., t_{n-1}$ are terms,
then $\mathcal{M}, f \models Rt_0...t_{n-1}$ iff$_{\text{Def}}$ $I(R)(I^f(t_0), ..., I^f(t_{n-1})) = 1$.

iii. It is never the case that $\mathcal{M}, f \models \bot$.

iv. If $A = \neg B$, then $\mathcal{M}, f \models A$ iff$_{\text{Def}}$ it is not the case that $\mathcal{M}, f \models B$.

v. If $A = B \wedge C$, then $\mathcal{M}, f \models A$ iff$_{\text{Def}}$ $\mathcal{M}, f \models B$ and $\mathcal{M}, f \models C$.
If $A = B \vee C$, then $\mathcal{M}, f \models A$ iff$_{\text{Def}}$ $\mathcal{M}, f \models B$ or $\mathcal{M}, f \models C$.
If $A = B \to C$, then $\mathcal{M}, f \models A$ iff$_{\text{Def}}$ $\mathcal{M}, f \models C$ or it is not the case that $\mathcal{M}, f \models B$.

vi. If $A = \forall x(B)$, then $\mathcal{M}, f \models A$ iff$_{\text{Def}}$ for all $g : \mathcal{V} \to M$ that agree with f on the assignment of all variables except possibly x, $\mathcal{M}, g \models B$.

If $A = \exists x(B)$, then $\mathcal{M}, f \models A$ iff$_{\text{Def}}$ there is a $g : \mathcal{V} \to M$ that agrees with f on the assignment of all variables except possibly x and $\mathcal{M}, g \models B$.

Remark: These clauses for the quantifiers are *first order*, as we quantify over *things* in the domain. A *second order* quantifier would quantify over *properties* (or, equivalently, sets of things). For example, in second order logic one can express something like 'there is a property that the thing x has' without specifying an explicit property, like in first order logic one can say 'there is a thing that has the property P' without specifying an explicit thing. Higher order logic can quantify over properties of properties, and then their properties *etc.*

Instead of $\mathcal{M}, f \models A$ we sometimes also write $M, I, f \models A$, if it is important to highlight the function I.

For an example, again consider the language of arithmetic and its standard interpretation $\langle \mathbb{N}, I \rangle$. Let $f : \mathcal{V} \to \mathbb{N}$ be any assignment function.

- $\mathbb{N}, I, f \models +ss0sss0 \equiv sssss0$ (i.e. $2 + 3 = 5$).
- $\mathbb{N}, I, f \models \forall x \neg 0 \equiv sx$ (i.e. there is no number whose successor is 0).
- $\mathbb{N}, I, f \models \forall x(\neg x \equiv 0 \to \exists y(x \equiv sy \land \forall z(x \equiv sz \to y \equiv z))$
  (i.e. every number except 0 has a unique predecessor).

We are usually not particularly interested in formulae with free variables. The assignment function is intuitively tangential to what is "going on" in the structure. We more or less only keep the assignment functions around because they help us define the satisfaction-conditions of the quantifiers. There is however an exception: using formulae with free variables, we can define relations that we may have not added to our initial language. For example

- The following formula $L(x, y)$ defines when x is strictly less than y in the language of arithemtic:

  $$L(x, y) = \exists z(\neg z \equiv 0 \land +xz \equiv y).$$

  Observe that in the intended interpretation of arithmetic, for each assignment function f, $\mathbb{N}, I, f \models L(x, y)$ iff $f(x) < f(y)$.

As said, models are to predicate logic what valuations are to propositional logic. Accordingly, we can use them to define semantic consequence.

Definition 42 (Semantic Consequence)**.** Let $\mathcal{L}$ be a language, let $\Gamma \subseteq \texttt{wff}(\mathcal{L})$ and $A \in \texttt{wff}(\mathcal{L})$. Define $\Gamma \models A$ iff$_{\text{Def}}$ for all $\mathcal{L}$-structures $\mathcal{M}$ and all assignments f, if $\mathcal{M}, f \models B$ for all $B \in \Gamma$, then also $\mathcal{M}, f \models A$.

We can now define our usual notions.

Definition 43. Let $\mathcal{L}$ be a language and $\mathcal{A}$ be a well-formed formula in $\mathcal{L}$.

- A is a **tautology** iff$_{\text{Def}}$ for all $\mathcal{L}$-structures $\mathcal{M}$ and all assignments $f : \mathcal{V} \to M$, $\mathcal{M}, f \models A$.
- A is a **satisfiable** iff$_{\text{Def}}$ there is a $\mathcal{L}$-structure $\mathcal{M}$ and an assignment $f : \mathcal{V} \to M$ such that $\mathcal{M}, f \models A$.
- A and B are **equivalent** iff$_{\text{Def}}$ for all $\mathcal{L}$-structures $\mathcal{M}$ and assignments $f : \mathcal{V} \to M$, it is the case that $\mathcal{M}, f \models A$ iff $\mathcal{M}, f \models B$.

If A is a tautology, we also write $\models A$ (which indeed means the same as $\emptyset \models A$).

We can again characterise equivalence in the object language and by way of semantic consequence.

Theorem 3.2. *Let $\mathcal{L}$ be a language and A and B be wffs in $\mathcal{L}$. A and B are equivalent iff $(A \to B) \wedge (B \to A)$ is a tautology.*

Theorem 3.3. *Let $\mathcal{L}$ be a language and A and B be wffs in $\mathcal{L}$. A and B are equivalent iff $A \models B$ and $B \models A$.*

The proofs are immediate consequences of the definition of $\models$.

Remark: Some logicians use a different definition of semantic consequence. Call a formula A *true* in a structure $\mathcal{M}$ iff$_{\text{Def}}$ for all assignments f, $\mathcal{M}, f \models A$. Then one may define $\Gamma \models A$ iff all models $\mathcal{M}$ that make all members of Γ true, also make A true. A side-effect is that a formula $A(x)$ is equivalent to $\forall x A$. One can also not define semantic consequence for formulae with free variables at all and instead say that implicitly all free variables are universally quantified. **We will not do any of this.** But this is a reminder that when free variables are concerned, there is some variation in the literature and you should **always carefully look at the definitions**. As formulae with free variables are very rarely interesting when talking about semantic consequence, this will not concern us much. All the different definitions agree on when $\Gamma \models A$ is the case when neither A nor any member of Γ have free variables.

Some important equivalences are (in all languages):

1. $\forall x A$ is equivalent to $\neg \exists x \neg A$.
2. $\exists x A$ is equivalent to $\neg \forall x \neg A$.
3. $\forall x (A \wedge B)$ is equivalent to $\forall x A \wedge \forall x B$.
4. $\exists x (A \vee B)$ is equivalent to $\exists x A \vee \exists x B$.

The proof of the first example is as follows.

Proof. Let $\mathcal{L}$ be a language, A be a formula, $\langle M, I \rangle$ be a $\mathcal{L}$-structure and f be an assignment.

Left-to-right. By contraposition. Assume that not $M, I, f \models \neg \exists x \neg A$ and show that not $M, I, f \models \forall x A$. By the clause for negation, the assumption entails that $M, I, f \models \exists x \neg A$. This means that there is an assignment g differing from f at most in what it assigns to x so that $M, I, g \models \neg A$. But this means that not $M, I, g \models A$. This means that it is not the case that for all assignments g that differ from f at most in what they assign to x it is the case that $M, I, g \models A$. Thus by definition it is not the case that $M, I, g \models \forall x A$.

Right-to-left. By contraposition. Assume that not $M, I, f \models \forall x A$ and show that not $M, I, f \models \neg \exists x \neg A$. The assumption entails that there is an assignment g differing from f at most in what it assigns to x so that not $M, I, g \models A$, i.e. $M, I, g \models \neg A$. Thus by definition, $M, I, g \models \exists x \neg A$. Hence it is not the case that $M, I, g \models \neg \exists x \neg A$. □

Note however that the following are **not equivalent**.

1. $\forall x(A \vee B)$ is not equivalent to $\forall xA \vee \forall xB$.

2. $\exists x(A \wedge B)$ is not equivalent to $\exists xA \wedge \exists xB$.

For a counterexample , consider a language with two constant symbols c_0 and c_1. Let $\langle M, I\rangle$ be a structure with $M = \{0, 1\}$ and $I(c_0) = 0$ and $I(c_1) = 1$.

Then clearly for all assignments f it is the case that $M, I, f \models \forall x(x \equiv c_0 \vee x \equiv x_1)$, but not the case that $M, I, f \models \forall x(x \equiv c_0) \vee \forall x(x \equiv c_1)$. This shows (1).

Also, for all assignments f it is the case that $M, I, f \models \exists x(x \equiv c_0) \wedge \exists x(x \equiv c_1)$, but not the case that $M, I, f \models \exists x(x \equiv c_0 \wedge x \equiv c_1)$. This shows (2).

Now, as usual, we can lift the definition of satisfiability to sets of formulae.

Definition 44. Let $\mathcal{L}$ be a language and Γ be a set of well-formed formulae in $\mathcal{L}$.

- Γ is **satisfiable** iff$_{\text{Def}}$ there is a $\mathcal{L}$-structure $\mathcal{M}$ and an assignment $f : \mathcal{V} \to M$ such that for all $A \in \Gamma, \mathcal{M}, f \models A$.

And then we have our usual theorem about semantic consequence and satisfiability.

Theorem 3.4. *Let $\mathcal{L}$ be a language, Γ be a set of wffs in $\mathcal{L}$ and $\mathcal{A}$ be a wff in $\mathcal{L}$. Then: $\Gamma \models A$ iff $\Gamma \cup \{\neg A\}$ is not satisfiable.*

The proof is analogous to the proof of Theorem 2.13.

We do not, however, have something analogous to *Coinciding Valuations* (Theorem 2.4). Using this theorem, we could check whether a formula is satisfiable by only checking what a finite amount of valuations assigns to a finite amount of atoms. The analogue would be that one could check whether a predicate logic formula A is satisfiable by checking a finite amount of models with finite domain. But some formulae in predicate logic can only be satisfied by infinite models.

An example are *Robinson's axioms.* Let $\mathcal{L}^A$ be the language of arithmetic (containing a constant symbol 0,unary function s and binary functions $+$, $\cdot$).

Definition 45. Robinson's axioms are the set of the following formulae in $\mathcal{L}^A$.

1. $\forall x(\neg sx \equiv 0)$.

2. $\forall x \forall y(sx \equiv sy \to x \equiv y)$.

3. $\forall x(x \equiv 0 \vee \exists y(sy = x))$.

4. $\forall x(x + 0 \equiv x)$ and $\forall x \forall y(x + sy \equiv s(x + y))$.

5. $\forall x(x \cdot 0 \equiv 0)$ and $\forall x \forall y(x \cdot sy \equiv (x \cdot y) + x)$.

It is easy to see that Robinson's axioms are only all satisfied in infinite models. To wit: Let for every natural number n, 's^n0' be the term obtained by prefixing 0 with n many s and (in particular, $s^00 = 0$). The intuitive reason is that for all numbers n and m, if $n \neq m$, then for every model of Robinson's axioms M, I, f, $I^f(s^m0) \neq I^f(s^n0)$, so M has at least as many members as there are natural numbers. Formally:

Theorem 3.5. *If M, I, f is a model of all Robinson axioms, then for all n and $m > 0$, if $m < n$, then $I^f(s^m0) \neq I^f(s^n0)$.*

Proof. Let M, I, f be a model of all Robinson axioms. If $n = 0$ there are no $m < n$, so there is nothing to show. Thus fix some arbitrary $n > 0$ and $m < n$. Assume for *reductio* that $I^f(s^n0) = I^f(s^m0)$. We will then show that $I^f(0) = I^f(s^{n-m}0)$, contradicting the first Robinson axiom.

Formally this goes as follows. Show by induction that for all k, if $k \leq m$, then $I^f(s^{m-k}0) = I^f(s^{n-k}0)$. (We will then get the desired contradiction in the case $k = m$.)

- Base $k = 0$. Then $I^f(s^{m-k}0) = I^f(s^m0) = I^f(s^n0) = I^f(s^{n-k}0)$ by the *reductio* assumption.
- Inductive step. Induction hypothesis: assume that for some fixed but arbitrary k, if $k \leq m$, then $I^f(s^{m-k}0) = I^f(s^{n-k}0)$.

  It is to show that if $k + 1 \leq m$, then $I^f(s^{m-(k+1)}0) = I^f(s^{n-(k+1)}0)$. If $k + 1 > m$ there is nothing to show, so assume that $k + 1 \leq m$. This entails that $k < m$, so from the IH we get that $I^f(s^{m-k}0) = I^f(s^{n-k}0)$.

  Because $k < m$, we can write $s^{m-k}0$ as $ss^{m-(k+1)}0$ and because $m < n$ also write $s^{n-k}0$ as $ss^{n-(k+1)}0$. So by the IH, it follows that $I^f(ss^{m-(k+1)}0) = I^f(ss^{n-(k+1)}0)$.

  By definition of $\models$, this means that $M, I, f \models ss^{m-(k+1)}0 \equiv ss^{n-(k+1)}0$. By the second Robinson axiom, it follows that $M, I, f \models s^{m-(k+1)}0 \equiv s^{n-(k+1)}0$. Thus by definition of $\models$, it is the case that $I^f(s^{m-(k+1)}0) = I^f(s^{n-(k+1)}0)$, which was to show.

This concludes the induction. Thus for all $k \leq m$, $I^f(s^{m-k}0) = I^f(s^{n-k}0)$.

But now consider the case $k = m$. Then $0 = s^{m-k}0$ and $s^{n-k}0 = s^{n-m)}0$, which means by the above result that $I^f(0) = I^f(s^{n-m}0)$. Because $m < n$, we can write the latter as $I^f(ss^{n-(m+1)}0)$. Then $M, f, I \models 0 \equiv ss^{n-(m+1)}0$. This contradicts the first Robinson axiom.

Hence by *reductio*, $I^f(s^m0) \neq I^f(s^n0)$. □

Thus, there is no finite model of the conjunction of the Robinson axioms. So to verify that they are satisfiable, we need to take an infinite model (the natural numbers will do) and *show* by a proof that they are a model. But this requires cleverness, not just a mechanical procedure in which one checks finitely many finite assignments of truth values.

3.5 Substitution

It is, in some sense, obvious that it does not matter which particular variables we use in a formula. For example, both $Q(x) = \exists y(\cdot yy \equiv x)$ and $Q'(z) = \exists x(\cdot xx \equiv z)$ formalise (in the intended interpretation of arithmetic) the property of *being a square*. So we may *substitute* the variables occurring in a formula for other variables without changing anything about the meaning of the formula. But we need to be careful. In $Q(x)$ we cannot just replace x with y because the result would be $\exists y(\cdot yy \equiv y)$ which is just a tautology.

We may indeed want to substitute entire *terms* in a formula. For example, we may want to define the property $x_1 + x_2$ *is a square* (a formula in two free variables) by substituting the term $+x_1x_2$ in Q for x. The result is $\exists y(\cdot yy \equiv +x_1x_2)$, which indeed does what we want. But again we need to be careful. If we were to substitute $+xy$ for x in Q, we would obtain $\exists y(\cdot yy \equiv +xy)$, which again says something else.

After some inspection of these problems, we see that they arise when we substitute a variable bound by a quantifier. Thus we need to be careful about bound variables when defining substitution. First define substitution in terms.

Definition 46 (Substitution in terms)**.** Let $\mathcal{L}$ be a language, s and t be terms in $\mathcal{L}$ and x be a variable. The **substitution of** t **for** x **in** s, written as $s[t/x]$ is defined by the following recursion.

- If $s = y$ where y is a variable, then $s[t/x] = t$ iff $x = y$ and s otherwise.
- If $s = c$ where c is a constant symbol, then $s[t/x] = c$.
- If $s = Ft_0...t_{n-1}$ for an n-ary function symbol F and terms $t_0, ..., t_{n-1}$, then $s[t/x] = Ft_0[t/x]...t_{n-1}[t/x]$.

Note that this definition does not ensure that substitutions can be inverted. For example, $(x + y)[y/x]$ is $(y + y)$. No substitution will get us back to $x + y$ and in particular $(x + y)[y/x][y/x]$ is $x + x$. So it is possible to irretrievably lose some information when substituting. While this sounds odd, it is something that we want to be able to do (as will become clear later).

The following definition takes care of our problems with quantifiers.

Definition 47 (Substitution)**.** Let $\mathcal{L}$ be a language, x be a variable, t be a term in $\mathcal{L}$ and A be a wff in $\mathcal{L}$. The **substitution of** t **for** x **in** A, written as $A[t/x]$, is defined by the following recursion.

- If $A = \bot$, then $A[t/x] = A$.
- If $A = t_0 \equiv t_1$, then $A[t/x] = t_0[t/x] \equiv t_1[t/x]$.
- If $A = Rt_0...t_{n-1}$ for an n-ary relation symbol R and terms $t_0, ..., t_{n-1}$, then $A[t/x] = Rt_0[t/x]...t_{n-1}[t/x]$.
- If $A = \neg B$, then $A[t/x] = \neg B[t/x]$.
- If $A = B \wedge C$, then $A[t/x] = B[t/x] \wedge C[t/x]$.
- If $A = B \vee C$, then $A[t/x] = B[t/x] \vee C[t/x]$.
- If $A = B \to C$, then $A[t/x] = B[t/x] \to C[t/x]$.

- If $A = \forall y B$, then if $y = x$ or y occurs in t, let y' be a variable symbol that occurs neither in A nor in t, let $B' = B[y'/y]$ and define $A[t/x] = \forall y'(B'[t/x])$. Else define $A[t/x] = \forall y(B[t/x])$.
- If $A = \exists y B$, then if $y = x$ or y occurs in t, let y' be a variable symbol that occurs neither in A nor in t, let $B' = B[y'/y]$ and define $A[t/x] = \exists y'(B'[t/x])$. Else define $A[t/x] = \exists y(B[t/x])$.

This means that we do the obvious thing every time, except that in quantifiers we take care to rename all bound variables to a variable that does not occur in the rest of the formula or in the term we substitute. It is obviously the case that renaming bound variables is harmless.

Theorem 3.6. *Let $\mathcal{L}$ be a language, A a wff in $\mathcal{L}$ and x and y be variables such that y does not occur in A. Then $\forall x A$ is equivalent to $\forall y(A[y/x])$.*

The proof is immediate from the definition of $\models$.

Note that Definition 47 is nevertheless somewhat problematic: in the quantifier cases, we just 'pick' an unused variable symbol y', but we do not state how to find this y' (we use the Axiom of Choice here). Thus if someone computes the substitution $(\forall y B)[t/y]$ by replacing y with y' and someone else does this by replacing y with y'' and $y' \neq y''$, both have equal claim to be correct.

One possible way out of this is that instead of appealing to Choice, we can define a specific variable that we are substituting. E.g. we can recall that we enumerated the variables as $\mathcal{V} = \{x_i \mid i \in \mathbb{N}\}$ and define the quantifier cases as follows.

- If $A = \forall y B$, then if $y = x$ or y occurs in t, let x_i be the variable symbol where i is minimal with the property that x_i occurs neither in A nor in t, let $B' = B[y'/y]$ and define $A[t/x] = \forall y'(B'[t/x])$. Else define $A[t/x] = \forall y(B[t/x])$.

  (existential quantifier analogous)

Then we have defined unique and computable substitutions. But this is cumbersome in practice. A more 'pragmatic' solution is to say that if we are in a case where we have to rename, we just don't do a substitution. To be more formal, consider the following definition.

Definition 48. Let $\mathcal{L}$ be a language, x be a variable, t be a term in $\mathcal{L}$ and A be a wff in $\mathcal{L}$. We say that t is **free for** x **in** A iff there is no quantifier binding x in A and whenever $\forall y B$ or $\exists y B$ is a subformula of A and x occurs in B, then y does not occur in t.

Basically, t is free for x in A if replacing (without renaming anything) t for x in A does not result in a variable in t being bound by a quantifier. Some examples:

1. x_1 is free for x_0 in $\exists x_2 R(x_2, x_0)$.
2. x_0 is free for x_0 in $\exists x_2 R(x_2, x_0)$.
3. x_2 is not free for x_0 in $\exists x_2 R(x_2, x_0)$.
4. $x_0 + x_1$ is free for x_0 in $\exists x_2 R(x_2, x_0)$.

5. $x_0 + x_2$ is not free for x_0 in $\exists x_2 R(x_2, x_0)$.

Note that it is permissible that x occurs in t for t to be free for x in A.

Then the following is directly provable from Definition 47.

Lemma 3.7 (Direct Substitution)**.** *Let $\mathcal{L}$ be a language, x be a variable, t be a term in $\mathcal{L}$ and A be a wff in $\mathcal{L}$ such that t is free for x in A. Then $A[t/x]$ is the result of replacing all occurrences of x in A with t.*

We will in practice only use direct substitutions, but it is useful to have the general definition in place to prove theorems about substitution in general.

Sometimes we will want to make multiple substitutions, e.g. $A[t_0/x][t_1/y]$. This may have an unintended side-effect: if y occurs in t_0, then in this formula, we will have also substituted t_1 in t_0. Sometimes we do not want to do this. In this case we write $A[t_0, t_1/x, y]$ for the simultaneous substitution of t_0 for x and t_1 for y (without substituting occurrences of x or y in t_0 or t_1).

In general, we write $A[t_0, ..., t_{n-1}/x_0, ..., x_{n-1}]$ for the formula obtained by *simultaneously* substituting t_i for x_i. Defining simultaneous substitution is analogous to the definition of simple substitution above (we will not require a formal definition, as we will below see how we can obtain simultaneous substitution from iterated simple substitution).

3.6 Natural Deduction for Predicate Logic

We obtain the Natural Deduction calculus for predicate logic by extending the Natural Deduction calculus for propositional logic with (a) rules for quantifiers and (b) rules for identity. We fix a language $\mathcal{L}$ and define our calculus as follows.

The following are the rules for the universal quantifier $\forall$. There is something new: we indicate next to each rule under **which conditions** it can be applied.

$$(\forall\text{I.})\ \frac{A[y/x]}{\forall x A}\ \text{if } y \text{ is a variable not occurring free in } \forall x A \text{ or in premisses or undischarged assumptions used to derive } A[y/x]. \qquad (\forall\text{E.})\ \frac{\forall x A}{A[t/x]}\ \text{where } t \text{ is any term.}$$

The intuitive justification for the Elimination rule is easy: if we have derived that everything is A, then for every name we have for a thing—i.e. for every term—we have that it is A. The Introduction rule is perhaps a bit more difficult to justify.

One's intuitive idea for an Introduction rule may be that if we have derived for all things that they are A, we may infer $\forall x A$. However, there are two problems with this: (i) we may have infinitely many things (but proofs are finite), so we might not be able to say that all things are A; (ii) we may not not have a name for every thing and for this reason be unable to say that all things are A.

A better idea is the following. If we derive for an *arbitrary name* that it has property A, then all things have property A, so we may infer $\forall x A$. This is a version of the proof method we have been using so far to prove universally quantified statements.

Proving Universals

To prove that all x are P take a fixed but arbitrary y and show that it has P.

A free and unused variable y is the formal version of 'fixed but arbitrary'. It is fixed, because under every assignment, y gets a definitive value. And if y occurs in a premiss or undischarged assumption, then we may infer some particular properties of y—which means that y is not arbitrary.

In addition, y cannot occur free in $\forall x A$ because we want to ensure that from some unused y having property A, (∀I.) derives that everything has the property A. If y occurs free in $\forall x A$, this is not necessarily the case. For example, if $A = x \equiv y$, then $A[y/x] = y \equiv y$. But from $y \equiv y$ it should not follow that $\forall x(x \equiv y)$, i.e. that all things are equal to y.

If y occurs *bound* in $\forall x A$ and $y \neq x$, then in $A[y/x]$, the bound instances will have been renamed; this is formally harmless, but in practice annoying. So when applying (∀I.) we will always pick either x itself or a y that has not been used *at all*—that is the easiest and most safe option.

The following are the rules for the existential quantifier ∃.

$$(\exists\text{I.})\ \frac{A[t/x]}{\exists x A}\ \text{where } t \text{ is any term.} \qquad (\exists\text{E.})^i\ \frac{\exists x A \qquad \begin{array}{c}[A[y/x]]^i \\ \vdots \\ B\end{array}}{B}\ \begin{array}{l}\text{if } y \text{ does not occur free in } \exists x A,\ B, \text{ or premisses} \\ \text{or undischarged assumptions.}\end{array}$$

Here, the Introduction rule is clear: if we know about any particular thing that it is A, we may infer that there is something that is A. The Elimination rule may again seem strange.

Intuitively, one may want to eliminate existential quantifiers as follows. If we have $\exists x A$ we know that there is one thing with A. Now, if we can show for everything that if it is A, then some B is true, we can conclude that B is the case (this is like a general version of Proof by cases). But again we are hindered by the fact that proofs are finite and that we may not have names for everything.

So we use the same plan as for the Introduction for universal quantifiers. If we know that something is A and from the assumption that *an arbitrary name* has property A we can infer that B, then we are entitled to conclude that B. Again, this is a proof method we have already used.

Proving from Existentials

If we know that there is an x with P and we want to show that B is the case, let y be arbitrary but fixed with property P and show that B.

Usually, when we used this method, B was a contradiction.

This explains why in (∃E.), y may not occur free in premisses or assumptions—as above, these are requirements for y being arbitrary. In addition, y may not occur free in $\exists x A$ because if it would, then the property $A(y)$ is not the property that $\exists x A$ says one x possesses (as it was above for the universal quantifier). It may also not occur in the conclusion B because we want to show that B is the case *regardless* of what y denotes, so B cannot itself mention y.

This suggests that the rules for the Introduction of universal quantifiers and the Elimination of existential quantifiers are tightly related. Indeed, the restrictions on (∃E.) can also be motivated by *formally deriving* them from (∀I.). This goes as follows. We know that (in the semantics) $\exists x A$ and $\neg\forall x\neg A$ are equivalent. We can reproduce this in the proof theory. To wit, if we treat $\exists x A$ as an abbreviation of $\neg\forall x\neg A$, we can derive (∃E.) as follows.

$$\dfrac{\dfrac{\neg\forall x\neg A \qquad \dfrac{\dfrac{\dfrac{[\neg B]^1 \qquad \dfrac{\begin{matrix}[A[y/x]]^2\\ \vdots\\ B\end{matrix}}{A[y/x]\to B}\,(\to\text{I.})^2}{\neg(A[y/x])}\,(\text{Contraposition})}{(\neg A)[y/x]}\,(\text{Substitution})}{\forall x\neg A}\,(\forall\text{I.})}{\bot}\,(\neg\text{E.})}{B}\,(\text{RAA})^1$$

Note that in the step labelled as ‘Substitution’, the formulae above and below the line are the *same* formula (literally, the same sequence of symbols). So this does not mean that there is a rule called ‘Substitution.’ The proof given here is a scheme and the ‘Substitution’ step is there to signpost that according to the definition of substitution, we can apply (∀I.) to the conclusion of (Contraposition). In any *concrete* formal proof there would not be a step there at all.

Now observe that when we apply (∀I.), $\neg B$ is an undischarged assumption, so y must not be occuring free in B (and other premisses or assumptions) for this proof to be correct. The conditions for (∀I.) also demand that y be not free in $\forall x A$, so it must not be free in $\neg\forall x\neg A$ either.

We can also derive (∃I.) from (∀E.) if we abbreviate $\exists x A$ as $\neg\forall x\neg A$.

$$\dfrac{\dfrac{A[t/x] \qquad \dfrac{\dfrac{\forall x\neg A}{\neg A[t/x]}\,(\forall\text{E.})}{\neg(A[t/x])}\,(\text{Substitution})}{\bot}\,(\neg\text{E.})}{\neg\forall x\neg A}\,(\neg\text{I.})^1$$

As a side benefit, these arguments establish that we could just abbreviate ∃ by ∀ and ¬.

Finally, the following rules govern identity.

$$(\equiv\text{R})\,\dfrac{}{t\equiv t}\ \text{where } t \text{ is a term} \qquad (\equiv\text{S})\,\dfrac{t_0\equiv t_1}{t_1\equiv t_0}\ \text{where } t_0 \text{ and } t_1 \text{ are terms} \qquad (\equiv\text{T})\,\dfrac{t_0\equiv t_1 \qquad t_1\equiv t_2}{t_0\equiv t_2}\ \text{where } t_0, t_1 \text{ and } t_2 \text{ are terms}$$

These formalise the basic properties that our meta-language $=$ has: reflexivity, symmetry and transitivity. We add to these a rule stating that identicals are intersubstitutable.

$$(\equiv\text{Sub})\,\dfrac{t_0\equiv t_1 \qquad A[t_0/x]}{A[t_1/x]}\ \text{where } t_0 \text{ and } t_1 \text{ are terms and } x \text{ is a variable}$$

These rules for identity are somewhat inelegant; certainly they do not fit into a schema for Introduction or Elimination. It is possible to give I/E rules for identity, but these rules are more difficult to handle.

These rules are also more complex than they would strictly need to be. We could state them for variables instead of terms and derive the versions with terms by introducing and eliminating universal quantifiers. While this is a technical possibility, we don't do so here because it would be odd if the fact that, for example, a term is self-identical has anything to do with the meaning of the quantifiers.

Sometimes, we may not want to substitute *all* occurrences of a variable in a formula in (≡Sub). For instance, maybe we have $t \equiv t$ and we want to derive from this that $\exists x(x \equiv t)$. We get such 'partial' substitutions from our quantifier rules. Note that $t \equiv t$ can be written as $(x \equiv t)[t/x]$. (If you recall a previous remark, this is why it is *good* for us that substitutions can 'lose information'). This means that the following is a correct proof.

$$\frac{t \equiv t}{\exists x(x \equiv t)}\ (\exists\text{I.})$$

A more complex example is that maybe from $x \equiv y$ and $x + x \equiv y$, we want to infer that $x + y \equiv y$, i.e. only replacing one occurrence of x by y. This follows immediately from (≡Sub). To see this, note that $x + y \equiv y$ is the same term as $(x + z \equiv y)[y/z]$ and $x + x \equiv y$ is the same term as $(x + z \equiv y)[x/z]$. Thus the following is a correct proof.

$$\frac{x \equiv y \qquad x + x \equiv y}{x + y \equiv y}\ (\equiv\text{Sub})$$

It is then easy (but tedious) to prove that the following general version of ≡-Substitution holds for any formula A and n many variables $x_0, ...x_{n-1}$.

$$(\equiv\text{Sub*})\ \frac{t_0 \equiv s_0 \qquad t_1 \equiv s_1 \qquad \dots \qquad t_{n-1} \equiv s_{n-1} \qquad A[t_0, t_1, ..., t_{n-1}/x_0, x_1, ..., x_{n-1}]}{A[s_0, s_1, ..., s_{n-1}/x_0, x_1, ..., x_{n-1}]}$$

We do this by applying (≡Sub) n-times and in step i we leave out any occurrences of x_i in all previous s_j (i.e. where $j < i$). This is also how we can perform simultaneous substitutions in proofs.

3.7 Hilbert calculus

We can obtain a Hilbert-style calculus for predicate logic using the symbols $\equiv$, $\to$, $\neg$ and $\exists$ by letting our existing logical axioms be schematic for formulae in predicate logic. We then add to it the following logical axioms.

- For all terms t: $t \equiv t$.
- For all terms t_0, t_1: $t_0 \equiv t_1 \to t_1 \equiv t_0$.
- For all terms t_0, t_1, t_2: $(t_0 \equiv t_1 \land t_1 \equiv t_2) \to t_0 \equiv t_2$
- For all terms t_0, t_1 and variables x: $t_0 \equiv t_1 \to (A[t_0/x] \to A[t_1/x])$
- For all terms t and variables x: $A[t/x] \to \exists x A$.
- For all variables x: $(A \to B) \to (\exists x A \to B)$ **if** $x \notin \texttt{frvar}(B)$.

Although in propositional logic the Hilbert calculus substantially simplified some arguments, here it is not much more economical than Natural Deduction. The above logical axioms are just the results of applying (→I.) to the Natural Deduction rules. The only advantage is that we need not worry about dischargeable assumptions for defining the rules for the quantifiers.

It is straightforward to now prove the Deduction theorem *etc* for the extended calculus. But we do have to re-prove our meta-theorems to continue with the Hilbert calculus, so doing so would actually be more work than using the Natural Deduction calculus from here on out.

4 Soundness for Predicate Logic

Theorem 4.1 (Soundness of the First Order Predicate Calculus)**.** *Let $\mathcal{L}$ be a language, Γ be a set of $\mathcal{L}$-formulae and A be a $\mathcal{L}$-formula. If $\Gamma \vdash A$ then $\Gamma \models A$.*

We have basically seen already why the rules for the propositional logic connectives are sound: the clauses defining their satisfaction in a model essentially are the truth tables again. The difficult new case are the quantifier and identity rules. As these crucially involve substitutions, we need to prove some results about substitution.

Lemma 4.2 (Substitution in Terms)**.** *Let $\mathcal{L}$ be a language, M, I be an $\mathcal{L}$-structure, s and t be terms and x be a variable symbol. If f and g are assignments with $g(x) = I^f(t)$ and $g(v) = f(v)$ for all $v \neq x$, then $I^g(s) = I^f(s[t/x])$.*

Proof. Let $\mathcal{L}$ be a language, M, I be an $\mathcal{L}$-structure, x be a variable, t be a term and f and g be assignments with $g(x) = I^f(t)$ and $g(v) = f(v)$ for all $v \neq x$. Show by induction on the construction of terms that for all s it is the case that $I^g(s) = I^f(s[t/x])$.

- Base. If $s = c$ for a constant symbol c, then $I^g(s) = I(c)$ and $I^f(t[t/x]) = I(c)$, so $I^g(s) = I^f(s[t/x])$.
- Base. If $s = v$ is a variable, then we are in one of two cases.
    - Case 1. $v = x$. Then $s[t/x] = t$ by definition of substitution. So $I^g(s) = I^g(x) = I^f(s[t/x])$.
    - Case 2. $v \neq x$. Then $s[t/x] = s$ and $I^g(s) = g(v)$ which by assumption is equal to $f(v)$, which by definition is equal to $I^f(s[t/x])$.

  In either case, we conclude what is to show.
- Inductive step. Suppose that $s = Ft_0...t_{n-1}$ for an n-ary function symbol F. Induction hypothesis: for all $i < n$, $I^g(t_i) = I^f(t_i[t/x])$. It is to show that $I^g(s) = I^f(s[t/x])$.

  By definition of substitution on terms, $s[t/x] = Ft_0[t/x]...t_{n-1}[t/x]$. Thus by the definition of the interpretation of terms, $I^f(s[t/x]) = I(F)(I^f(t_0[t/x]), ..., I^f(t_{n-1}[s/x]))$. By the induction hypothesis, $I^f(s[t/x]) = I(F)(I^g(t_0), ..., I^g(t_{n-1}))$. By definition of the interpretation of terms, this just means that $I^f(s[t/x]) = I^g(s)$.

This concludes the induction. □

From this we get the analogous result about substituting in formulae. The intuitive meaning of the Substitution Lemma is that it doesn't matter whether the things we talked about are referred to by variable symbols or by terms. All that matters is *which* value they get from the interpretation and assignment.

Lemma 4.3 (Substitution Lemma)**.** *Let $\mathcal{L}$ be a language, M, I be an $\mathcal{L}$-structure, A be a $\mathcal{L}$-formula, x be a variable symbol, t be a term that is free for x in A. If f and g are assignments such that $g(x) = I^f(t)$ and for all variable symbols $v \neq x$, $f(v) = g(v)$, then $M, I, f \models A[t/x]$ iff $M, I, g \models A$.*

Proof. Let $\mathcal{L}$ be a language, M, I be an $\mathcal{L}$-structure x be a variable and t be a term. Show by induction on the construction of formulae that for all A and all assignments f and g with $g(x) = I^f(t)$ and $g(v) = f(v)$ for all $v \neq x$, if t is free for x in A, then it is the case that $M, I, f \models A[t/x]$ iff $M, I, g \models A$.

- Base: $A = t_0 \equiv t_1$. Let f and g be as required. By definition of substitution, $A[t/x] = t_0[t/x] \equiv t_1[t/x]$. Then: $M, I, f \models A[t/x]$ iff (def of $\models$) $I^f(t_0[t/x]) = I^f(t_1[t/x])$ iff (previous Lemma) $I^g(t_0) = I^g(t_1)$ iff (def of $\models$) $M, I, g \models A$.
- Inductive step negation. $A = \neg B$ for some formula B. Induction hypothesis: if f and g are as required and t is free for x in B, then $M, I, f \models B[t/x]$ iff $M, I, g \models B$. Show that if f and g are as required and t is free for x in A, then $M, I, f \models A[t/x]$ iff $M, I, g \models A$.

  So let f and g be as required (i.e. $g(x) = I^f(t)$ and $g(v) = f(v)$ for all $v \neq x$). If t is not free for x in A, there is nothing to show, so assume that it is. Then t is also free for x in B. Thus we can use the induction hypothesis to conclude that $M, I, f \models B[t/x]$ iff $M, I, g \models B$.

  Then: $M, I, f \models A[t/x]$ iff (def of substitution) $M, I, f \models \neg(B[t/x])$ iff (def of $\models$) not $M, I, f \models B[t/x]$ iff (contrapositive of the above) not $M, I, g \models B$ iff (def of $\models$) $M, I, g \models \neg B$ iff $M, I, g \models A$. This was to show.
- Inductive step universal quantifier. $A = \forall z B$ for some formula B and variable symbol z. Induction hypothesis: if f and g are as required and t is free for x in B, then $M, I, f \models B[t/x]$ iff $M, I, g \models B$. Show that if f and g are as required and t is free for x in A, then $M, I, f \models A[t/x]$ iff $M, I, g \models B$.

  So let f and g be as required. If t is not free for x in A, there is nothing to show, so assume that it is. Then t is also free for x in B, so we can use the induction hypothesis in the argument below. Because t is free for x in A, it is the case that $x \neq z$, so by definition of substitution, $A[t/x] = \forall z(B[t/x])$.

  Show left-to-right by contraposition. Assume not $M, I, g \models A$ and show that not $M, I, f \models A[t/x]$. By definition of $\models$, the assumption entails that there is an assignment h that only differs from g in what it assigns to z such that not $M, I, h \models B$. Define an assignment j by $j(z) = h(z)$ and $j(v) = f(v)$ for all $v \neq z$. Note the following.

  (a) Because t is free for x in A, z does not occur in t. So because j differs from f only on the value of z, it follows that $I^j(t) = I^f(t)$ and because $h(x) = g(x)$ it follows by definition of f and g that $I^j(t) = h(x)$.

(b) For all variables $v \neq z$ and $v \neq x$, $h(v) = g(v) = f(v) = j(v)$. And by definition of j, $h(z) = j(z)$. So for all $v \neq x$, $h(v) = j(v)$

Note that (a) and (b) mean that h and j are as required for the induction hypothesis. Thus, $M, I, h \models B$ iff $M, I, j \models B[t/x]$. Thus, as we have that not $M, I, h \models B$, it follows that not $M, I, j \models B[t/x]$. But by definition of j, j differs from f only on the value of z. So by definition of $\models$, this means that $M, I, f \not\models \forall z(B[t/x])$, i.e. $M, I, f \not\models A[t/z]$ by definition of substitution. This was to show.

The right-to-left direction is analogous.

- The cases where $A = \bot$, $A = B \vee C$, $A = B \wedge C$ or $A = B \to C$ are analogous to the negation case. The case where $A = \exists x B$ is analogous to the universal quantifier case and the case where $A = Rt_0 ... t_n$ is analogous to the identity case.

This concludes the induction. □

With this lemma in place, we have basically covered the difficult parts of the proof of Soundness.

We make a now familiar simplifying assumption. We have shown above that we may treat $\exists x A$ as an abbreviation of $\neg \forall x \neg A$ and we have seen in Theorem 2.21 that we can treat $\wedge$, $\vee$ and $\bot$ as abbreviations if we adopt the following rules.

$$(\to\text{I.})^i \; \dfrac{\begin{matrix}[A]^i \\ \vdots \\ B\end{matrix}}{A \to B} \qquad (\to\text{E.}) \; \dfrac{A \to B \qquad A}{B} \qquad (\text{NI}) \; \dfrac{A \to B \qquad A \to \neg B}{\neg A} \qquad (\text{DNE}) \; \dfrac{\neg\neg A}{A} \qquad (\text{E}) \; \dfrac{A \qquad B}{A}$$

This means that it suffices to show the following to establish Soundness. Let $\mathcal{L}$ be a language, Γ be a set of $\mathcal{L}$-formulae and A be a $\mathcal{L}$-formula such that A and all members of Γ do not contain the symbols $\bot$, $\vee$, $\wedge$ or $\exists$. Assume that $\Gamma \vdash A$ by a proof involving only the rules ($\to$I.), ($\to$E.), (NI), (DNE), (E), ($\forall$I.), ($\forall$E.), ($\equiv$R), ($\equiv$S), ($\equiv$T) and ($\equiv$Sub). Then show that $\Gamma \models A$. This suffices to show the full Soundness result, as any proof in the full language using all rules can be rewritten to a proof in the reduced language using the reduced set of rules.

As was the case for the proof of the soundness of the propositional calculus, it is not necessary to use such a reduction. This is merely a *convenience*: fewer rules means we have to check fewer inductive steps in the following proof.

Proof of Theorem 4.1. Let $\mathcal{L}$ be a language. We will prove the following by induction on the length of proofs:

For all sets of $\mathcal{L}$-formulae Γ and all $\mathcal{L}$-formulae A: if $\Gamma \vdash A$ by a proof of length n or less that involves only the reduced set of rules, then $\Gamma \models A$.

- Base: Suppose that $\Gamma \vdash A$ by a proof of length 1. This time, we are in one of two cases.
  - Case 1. A is a member of Γ. Then it is trivially the case that for all models $\mathcal{M}, f$, if for all $B \in \Gamma$, $\mathcal{M}, f \models B$, then also $\mathcal{M}, f \models A$, since A is itself a member of Γ. Thus $\Gamma \models A$ by definition of semantic consequence.

  - Case 2. The proof is a single application of (≡R). Then $A = t \equiv t$ for a term t. As then A is a tautology, it follows that $\Gamma \models A$.

- Inductive step. IH: Assume that for a fixed but arbitrary n it is the case that for all Γ and A, if $\Gamma \vdash A$ with a proof of length n or less that involves only the reduced set of rules, then also $\Gamma \models A$.

  It is to show that for all Γ and A where $\Gamma \vdash A$ by a proof of length $n + 1$ or less that involves only the reduced set of rules, it is the case that $\Gamma \models A$. So let Γ and A be arbitrary such that A follows from Γ by a proof tree of length $n + 1$ or less. If the length of the proof is n or less, we are done by the induction hypothesis. So assume it has length $n + 1$.

  We can do a case distinction on the last step in the proof of A.

  1. The last step is (DNE). Then $\Gamma \vdash \neg\neg A$ by a proof of length n. So by the IH $\Gamma \models \neg\neg A$. So for all models M, I, f such that $M, I, f \models B$ for all $B \in \Gamma$, it is also the case that $M, I, f \models \neg\neg A$.

     Let M, I, f be any such model. By definition of $\models$, $M, I, f \models \neg\neg A$ means that it is not the case that $M, I, f \models \neg A$. But this means that it is not the case that it is not the case that $M, I, f \models A$. But this just means that $M, I, f \models A$. As this goes for any M, I, f with $M, I, f \models B$ for all $B \in \Gamma$, it follows that $\Gamma \models A$ by the definition of semantic consequence.

  2. The last step is (→I.). Then $A = C \to D$ for some formulae C and D and there is a proof of length n showing that $\Gamma \cup \{C\} \vdash D$. By the induction hypothesis, $\Gamma \cup \{C\} \models D$. That is, for all models M, I, f such that $M, I, f \models B$ for all $B \in \Gamma$ and $M, I, f \models C$, it is also the case that $M, I, f \models D$.

     Let M, I, f be any model such that $M, I, f \models B$ for all $B \in \Gamma$. We may distinguish two cases

     - Case 1: $M, I, f \models C$. Then by the above, $M, I, f \models D$. Then by definition of $\models$, it is also the case that $M, I, f \models C \to D$, i.e. $M, I, f \models A$.
     - Case 2: it is not the case that $M, I, f \models C$. Then by definition of $\models$, it is also the case that $M, I, f \models C \to D$, i.e. $M, I, f \models A$.

     Thus $M, I, f \models A$. As this goes for any M, I, f with $M, I, f \models B$ for all $B \in \Gamma$, it follows that $\Gamma \models A$ by the definition of semantic consequence.

  3. The last step is (→E.). Then there is a formula C and proofs of C from Γ and of $C \to A$ from Γ. Both are shorter than the proof of A, so by the IH $\Gamma \models C \to A$ and $\Gamma \models B$. That is, for all models M, I, f such that $M, I, f \models B$ for all $B \in \Gamma$, it is also the case that $M, I, f \models C \to A$ and $M, I, f \models C$.

     Let M, I, f be any such model. By definition of $\models$, $M, I, f \models C \to A$ means that either $M, I, f \models A$ or it is not the case that $M, I, f \models C$. As it *is* the case that $M, I, f \models C$, it follows that $M, I, f \models A$. As this goes for any M, I, f with $M, I, f \models B$ for all $B \in \Gamma$, it follows that $\Gamma \models A$ by the definition of semantic consequence.

4. The last step is (NI). Then there is a C such that $A = \neg C$ and there are proofs showing that for some $D, \Gamma \vdash C \to D$ and $\Gamma \vdash C \to \neg D$. Both are shorter than the proof of A, so by the induction hypothesis, $\Gamma \models C \to D$ and $\Gamma \models C \to \neg D$. So for all models M, I, f such that $M, I, f \models B$ for all $B \in \Gamma$, it is also the case that $M, I, f \models C \to D$ and $C \to \neg D$.

   Let M, I, f be any such model. For *reductio*, assume that it is not the case that $M, I, f \models A$. Then $M, I, f \models C$. By definition of $\models$, $M, I, f \models C \to D$ means that either $M, I, f \models D$ or it is not the case that $M, I, f \models C$; and $M, I, f \models C \to \neg D$ means that either $M, I, f \models \neg D$ or it is not the case that $M, I, f \models C$. Since $M, I, f \models C$ by assumption, it follows that $M, I, f \models D$ and $M, I, f \models \neg D$. But this is impossible by definition of $\models$. Contradiction. Thus by *reductio*, $M, I, f \models A$. As this goes for any M, I, f with $M, I, f \models B$ for all $B \in \Gamma$, it follows that $\Gamma \models A$ by the definition of semantic consequence.

5. The last step is (E). Then there are proof trees showing that for some $B, \Gamma \vdash A$ and $\Gamma \vdash B$. Both are shorter than the proof of A we are considering in this step, so by the IH $\Gamma \models A$.

6. The last step is (∀I.). Then there is a C and variables x and y such that: (i) $A = \forall x C$, (ii) $\Gamma \vdash C[y/x]$ by a proof of length n; (iii) y does not occur in premisses and undischarged assumptions of this proof; (iv) y is not free in $\forall x C$. Let $\Gamma_0 \subseteq \Gamma$ be the set of premisses required for the proof of $C[y/x]$. By the induction hypothesis, $\Gamma_0 \models C[y/x]$. So for all models M, I, f such that $M, I, f \models B$ for all $B \in \Gamma_0$, it is also the case that $M, I, f \models C[y/x]$.

   Let M, I, f one of these models. We need to show that for all g that differ from f at most in what is assigned to x, it is the case that $M, I, g \models C$. So let g be an arbitrary such assignment.

   Note that we cannot conclude that $M, I, g \models C[y/x]$ because we can't know that M, I, g models all members of Γ_0. Instead, we exploit the fact that y is not free in the premisses.

   Define an assignment f' by $f'(y) = g(x)$ and $f'(v) = f(v)$ for all $v \neq y$. By (iii), y does not occur free in any $B \in \Gamma_0$, so the value assigned to y does not matter for their satisfaction. As for all $B \in \Gamma_0$ it is the case that $M, I, f \models B$ and f' differs from f only on the value of y, it follows that $M, I, f' \models B$ for all $B \in \Gamma_0$. As $\Gamma_0 \models C[y/x]$, it follows by the definition of semantic consequence that $M, I, f' \models C[y/x]$.

   Remark: what we used here is the 'arbitrariness' we coded in the rule for universal introduction: the assumptions of the proof don't decide anything about the value of y, so we can assign anything to y. We assign to y what g assigns to x. So we find a valuation that says that $C[y/x]$ is true for y having the value of $g(x)$. This *almost* says that g makes C true, but we need now to appeal to the second thing we coded in the rule: that $C[y/x]$ indeed expresses the same property as C.

   Now, by (iv), we are in one of two cases: either y is not free in C or $y = x$.

   - Case 1. y is not free in C. We may without loss of generality assume that y does not occur in C at all (if it occurs bound, just rename), so that y is free for x in t.

Define an assignment g' by $g'(y) = g(x)$ and $g'(v) = g(v)$ for all $v \neq y$. Note that for all $v \neq x$, it is the case that $g'(v) = f'(v)$ because f' differs from f only on y and g' differs from f only on x and y. Because both $g'(y)$ and $f'(y)$ are defined to be equal to $g(x)$, they only differ on x. But we know that $g'(x) = g(x)$ and $g(x) = f'(y)$, so $g'(x) = f'(y)$, which we can also write as $g'(x) = I^{f'}(y)$.

So we have that y is free for x in C, that $g'(x) = I^{f'}(y)$ and that $g'(v) = f'(v)$ for all $v \neq x$. Thus we can apply the Substitution Lemma to conclude from $M, I, f' \models C[y/x]$ that $M, I, g' \models C$. Because y is not free in C, the value of y does not matter for the satisfaction of C. So because g only differs from g' in what is assigned to y, it follows that $M, I, g \models C$.

– Case 2. $y = x$. Then $C[y/x] = C$ and $f' = g$, so $M, I, g \models C$.

In either case we conclude that $M, I, g \models C$. As g was arbitrary with the property of differing from f at most in the value of x, it follows that $M, I, f \models \forall x C$. As M, I, f was arbitrary, $\Gamma_0 \models \forall x C$, i.e. $\Gamma_0 \models A$.

Now consider any model M, I, g of all members of Γ. Then in particular M, I, g is also a model of all members of Γ_0. Thus $M, I, g \models A$. As M, I, g was arbitrary, $\Gamma \models A$.

7. The last step is ($\forall$E.). Then there is a formula C, a term t and a variable x such that $A = C[t/x]$ and there is a shorter proof of $A = \forall x C$. Without loss of generality we may assume that t is free for x in C (otherwise we just need to rename some bound variables in C). By the induction hypothesis, $\Gamma \models \forall x C$. Let M, I, f be an arbitrary model of all members of Γ, so $M, I, f \models \forall x C$. Then, by definition of $\models$, for all assignment functions f' that differ from f at most in what they assign to x, $M, I, f' \models C$.

   Define an assignment function g by $g(x) = I^f(t)$ and $g(v) = f(v)$ for all $v \neq x$. Note that g differs from f at most in what it assigns to x, so $M, I, g \models C$. By the Substitution Lemma, it follows that $M, I, f \models C[t/x]$. As M, I, f was arbitrary, it follows by definition of semantic consequence that $\Gamma \models C[t/x]$.

8. The last step is ($\equiv$S). Then there are terms t_0 and t_1 such that $A = t_1 \equiv t_0$ and there is a shorter proof of $t_0 \equiv t_1$. Let M, I, f be an arbitrary model of all members of Γ. By the induction hypothesis, $M, I, f \models t_0 \equiv t_1$, so by definition of $\models$, $I^f(t_0) = I^f(t_1)$. Then also by definition of $\models$, $M, I, f \models t_1 \equiv t_0$, i.e. $M, I, f \models A$. As M, I, f was arbitrary, $\Gamma \models A$.

9. The last step is ($\equiv$T). Then there are terms t_0, t_1 and t_1 such that $A = t_0 \equiv t_2$ and there are shorter proofs of $t_0 \equiv t_1$ and $t_1 \equiv t_2$. Let M, I, f be an arbitrary model of all members of Γ. By the induction hypothesis, $M, I, f \models t_0 \equiv t_1$ and $M, I, f \models t_1 \equiv t_2$, so by definition of $\models$, $I^f(t_0) = I^f(t_1)$ and $I^f(t_1) = I^f(t_2)$, so $I^f(t_0) = I^f(t_2)$. Thus by definition of $\models$, $M, I, f \models t_0 \equiv t_2$, i.e. $M, I, f \models A$. As M, I, f was arbitrary, $\Gamma \models A$.

10. The last step is ($\equiv$Sub). Then there are terms t_0 and t_1, a variable x and a formula B such that $A = B[t_1/x]$ and shorter proofs of $t_0 \equiv t_1$ and $B[t_0/x]$. Let M, I, f be an arbitrary model of

all members of Γ. By the induction hypothesis, $M, I, f \models t_0 \equiv t_1$ and $M, I, f \models B[t_0/x]$. In particular, this means that $I^f(t_0) = I^f(t_1)$.

Define an assignment function g by $g(x) = I^f(t_0)$ and $g(v) = f(v)$ for all $v \neq x$. By the Substitution Lemma, $M, I, g \models B$ iff $M, I, f \models B[t_0/x]$. Note that $g(x) = I^f(t_1)$, so by the Substitution Lemma, it is also the case that $M, I, g \models B$ iff $M, I, f \models B[t_1/x]$. Thus, $M, I, f \models B[t_0/x]$ iff $M, I, f \models B[t_1 x]$. And since the left-hand-side is true by the induction hypothesis, $M, I, f \models B[t_1/x]$. As M, I, f was arbitrary, $\Gamma \models B[t_1/x]$.

We have covered all cases, thus if $\Gamma \vdash A$ by a proof of length $n + 1$, then $\Gamma \models A$.

This concludes the induction. □

5 Completeness for Predicate Logic

Theorem 5.1 (Completeness of the First Order Predicate Calculus)**.** *Let $\mathcal{L}$ be a language, Γ be a set of $\mathcal{L}$-formulae and A be a $\mathcal{L}$-formula. If $\Gamma \models A$ then $\Gamma \vdash A$.*

Our strategy for proving this will be the same as in the propositional logic case. If we can show that every consistent set Γ is satisfiable, the completeness result follows. We can show that a consistent set is satisfiable by extending it to a maximally consistent set and then defining a model by the atomic formulae contained in the maximal set.

Henkin's Method

To prove completeness of some calculus with respect to some model theory:

1. Show that if every consistent set of formulae is satisfiable, completeness follows.
2. Show that every consistent set of formulae Γ is satisfiable by formalising a procedure that **extends** any consistent Γ to a set $\hat{\Gamma}$ that contains **sufficient syntactic information** to build a model out of syntax.

This isn't quite a proof method as we have seen so far (it is more vague than this). But it is a generally applicable recipe for proving completeness in basically any logic. For propositional logic, we saw that 'sufficient syntactic information' was provided by maximally consistent sets that 'decided' all the atoms (syntactically!). For predicate logic we need a little bit more cleverness.

5.1 Term Models

In propositional logic, all we needed was a valuation. We were able to obtain one by assigning the value 1 to all atoms in the maximally consistent extension. But in the predicate logic case, we need a *structure* consisting of a domain things and an interpretation of the language. Where would we take the domain from, when all we have is a set of formulae?

If all we have are terms and sentences, then the model must come from them. Our one and only choice is this: let the *terms themselves* form the things in the domain. Recall that for arithmetic, we had in the standard interpretation that the number n is denoted by the term $s^n 0$ (i.e. the nth successor of 0). In the language containing only the constant symbol 0 and the successor function s, we can define a *term model* as follows.

- Let the domain M be the set of all terms formed from 0 and s.
- $I(0) = 0$ (both occurrences of 0 are the *symbol*, not the number);
- Define the function $I(s) : M \to M$ by $I(s)(t) = s^\frown t$;
- Define an assignment f as $f(x) = x$ (as any variable is a term).

It is very straightforward to see that in this construction $I(t) = t$ for all terms. So in these constructions, the terms do double duty: they are syntactic objects in our language, but they *also* are the domain elements for the model we are constructing. (This has a whiff of *formalism*, the idea that mathematics can be done *only* by manipulating symbols, without bringing in anything that carries meaning beyond the symbols.)

We want to do this, or something similar, for *any* consistent set of formulae in any language. The following definition will help us.

Definition 49. Let $\mathcal{L}$ be a language.

- A set Γ of $\mathcal{L}$-formulae is said to **contain witnesses** iff$_{\text{Def}}$ for all formulae of the form $\forall x A$, there is a constant symbol c_A in the language such that if $\neg\forall x A \in \Gamma$, then also $\neg A[c_A/x] \in \Gamma$.
- A set Γ is called a **Henkin extension** in $\mathcal{L}$ iff$_{\text{Def}}$ it contains witnesses and is maximally consistent.

Containing witnesses means that if, according to Γ, there is a counterexample to an universally quantified claim, then there is also a *name* of a counterexample (a 'witness'). An equivalent and perhaps more obvious definition is that Γ contains witnesses iff whenever Γ contains $\exists x A$, it also contains $A[c_A/x]$. (We are not defining it like this because we will again treat $\exists$ as an abbreviation in the main argument.)

Remark: Oftentimes Γ is said to contains witnesses iff for all A, Γ contains the formula $\exists x A \to A[c_A/x]$, regardless of whether Γ contains a quantified statement saying that there should be a witness for A. The way presented here allows for a simplified argument below (Theorem 5.7).

The witnesses are important for the following reason. We want to show that every Henkin extension is satisfiable by constructing a model with a domain constructed from the terms. If we have a consistent set that does not contain witnesses, it could be that it contains an existential statement $\exists x A$, but there is no name for something that makes A true, i.e. no term in the language that makes $A[t/x]$ true. Then we cannot ensure that $\exists x A$ is true in models with domains constructed from terms. Having witnesses ensures that this is *not* the case.

As in the propositional logic case, maximally consistent sets of formulae are deductively closed.

Theorem 5.2. *Let $\mathcal{L}$ be a language and let Γ be a Henkin extension. Then for all A, if $\Gamma \vdash A$, then $A \in \Gamma$.*

Proof. Let $\mathcal{L}$ be a language and Γ be an arbitrary Henkin extension in $\mathcal{L}$. Let A be an arbitrary formula such that $\Gamma \vdash A$. Towards a *reductio*, assume that $A \notin \Gamma$. Because Γ is maximally consistent, this means that $\Gamma \cup \{A\}$ is inconsistent, i.e. $\Gamma \cup \{A\} \vdash \bot$. This means that by (¬I.), $\Gamma \vdash \neg A$. But because also $\Gamma \vdash A$, this means by (¬E.) that $\Gamma \vdash \bot$. Contradiction to the assumption that Γ is maximally consistent. Thus, by *reductio*, $A \in \Gamma$. □

The witnesses ensure that we have *enough* terms. We also need something parallel to ensure that we do not have *too many* terms in the domain of the term model. Unlike in the example above, we cannot just take the set of all terms to be the domain. This is because we might have two terms t and s that are not identical, but must receive the *same interpretation* because $t \equiv s$ is a member of Γ. For example, in the language of arithmetic, $s0$ and $+s00$ are distinct terms, but $s0 \equiv +s00$ is consistent with, say, the Robinson axioms. But if $s0$ and $+s00$ are distinct members of the domain, then $I(s0) = s0$ and $I(+s00) = +s00$, so $I(s0) \neq I(+s00)$, hence $s0 \equiv +s00$ will not come out as true in any such term model. The construction in the following theorem takes care of this issue.

We will avoid this issue as follows. Let $\mathcal{L}$ be a language and Γ be a Henkin extension in $\mathcal{L}$.

For every $\mathcal{L}$-term t consider the set $[t] = \{s \in \texttt{Tm}(\mathcal{L}) \mid t \equiv s \in \Gamma\}$. First show that for terms t and t' , if $t \equiv t' \in \Gamma$, then $[t] = [t']$.

Proof. Two sets are equal if they contain the same members, so we assume $t \equiv t' \in \Gamma$ and show that all members of $[t]$ are members of $[t']$ and *vice versa.*

- Let $s \in [t]$ be arbitrary. This means that $t \equiv s \in \Gamma$. Note that also $t \equiv t' \in \Gamma$, so by (≡S), $\Gamma \vdash t' \equiv t$. Note that by (≡T), $t' \equiv t, t \equiv s \vdash t' \equiv s$, so $\Gamma \vdash t' \equiv s$. Because Γ is deductively closed, $t' \equiv s \in \Gamma$. Thus $s \in [t']$.
- Let $s \in [t']$ be arbitrary. This means that $t' \equiv s \in \Gamma$. Note that also $t \equiv t' \in \Gamma$, so by (≡T), $\Gamma \vdash t \equiv s$. Because Γ is deductively closed, $t \equiv s \in \Gamma$. Thus $s \in [t]$.

Thus $[t] = [t']$. (Mathematically, what we have exploited here is that $t \equiv t' \in \Gamma$ defines an **equivalence relation** on the terms and the sets $[t]$ are **equivalence classes**.) □

With this in place, we may define our term model M, I, f as follows.

- $M = \{[t] \mid t \in \texttt{Tm}(\mathcal{L})\}$.
- If v is a variable, $f(v) = [v]$;
- If c is a constant symbol, $I(c) = [c]$;
- If F is an n-ary function symbol, $I(F) : M^n \to M$ is the function defined as follows: $I(F)([t_0], ..., [t_{n-1}]) = [Ft_0...t_{n-1}]$;
- If R is an n-ary relation symbol, $I(R) : M^n \to \{0, 1\}$ is the function defined as follows: $I(R)([t_0], ..., [t_{n-1}]) = 1$ iff $Rt_0...t_{n-1} \in \Gamma$.

Note that we are multiply-mentioning members in the definition of M, but this is harmless.

However, there is a potential for something to go wrong in this definition. We have defined the function $I(F)$ by saying that $I(F)([t_0], ..., [t_{n-1}] = [Ft_0...t_{n-1}]$, but suppose that we have terms $t'_0, ..., t'_{n-1}$ such that for all $i < n$, $[t_i] = [t'_i]$. Then $I(F)([t_0], ..., [t_{n-1}])$ and $I(F)([t'_0], ..., [t'_{n-1}])$ are the same function $I(F)$ applied to the same exact arguments as $[t_i]$ and $[t'_i]$ are just different ways of picking out the *same* member of M. So the result ought to be the same, i.e. it must be the case that $[Ft_0...t_{n-1}] = [Ft'_0...t'_{n-1}]$.

If this is *not* the case, then we have failed to define $I(F)$ to be a function (in mathematics, one would say that our definition was *not well-defined*). So we should prove that for all n-ary function symbols F and terms $t_0, .., t_{n-1}$, $t'_0, ..., t'_{n-1}$ such that $[t_0] = [t'_0]$, ..., $[t_{n-1}] = [t'_{n-1}]$ it is the case that $[Ft_0...t_{n-1}] = [Ft'_0...Ft'_{n-1}]$.

Proof. To show that $[Ft_0...t_{n-1}] = [Ft'_0...Ft'_{n-1}]$ we show that $Ft_0...t_{n-1} \equiv Ft'_0...Ft'_{n-1} \in \Gamma$. Recall the derived rule (≡Sub*).

$$(\equiv\text{Sub*})\ \frac{t_0 \equiv s_0 \qquad t_1 \equiv s_1 \qquad \ldots \qquad t_{n-1} \equiv s_{n-1} \qquad A[t_0, t_1, ..., t_{n-1}/x_0, x_1, ..., x_{n-1}]}{A[s_0, s_1, ..., s_{n-1}/x_0, x_1, ..., x_{n-1}]}$$

Let $x_0, ..., x_{n-1}$ be variable symbols and let $A = Ft_0...t_{n-1} \equiv Fx_0...x_{n-1}$. It is the case that:

1. $Ft_0...t_{n-1} \equiv Ft_0...t_{n-1} \quad = \quad A[t_1, ..., t_{n-1}/x_0, ..., x_{n-1}]$; and
2. $Ft_0...t_{n-1} \equiv Ft'_0...t'_{n-1} \quad = \quad A[t'_1, ..., t'_{n-1}/x_0, ..., x_{n-1}]$.

Note that by (≡R), $\Gamma \vdash Ft_0...t_{n-1} \equiv Ft_0...t_{n-1}$, i.e. $\Gamma \vdash A[t_1, ..., t_{n-1}/x_0, ..., x_{n-1}]$ by (1). Also note that from $[t_0] = [t'_0]$, ..., $[t_{n-1}] = [t'_{n-1}]$ it follows that for all $i < n$, $\Gamma \vdash t_i \equiv t'_i$. Thus we have all premisses of (≡Sub*), so $\Gamma \vdash A[t'_1, ..., t'_{n-1}/x_0, ..., x_{n-1}]$. But by (2), this means that $\Gamma \vdash Ft_0...t_{n-1} \equiv Ft'_0...t'_{n-1}$. Because Γ is deductively closed, $Ft_0...t_{n-1} \equiv Ft'_0...t'_{n-1} \in \Gamma$. This was to show. □

The same problem occurs for the definition of $I(R)$ for relation symbols R, but the proof that nothing goes wrong is analogous to the one for the function symbols. This means we have succeeded at defining a model.

Definition 50. Let $\mathcal{L}$ be a language and Γ be a Henkin extension in $\mathcal{L}$. The model M, I, f defined above is called the **term model** of Γ.

Remark: the term model does not have to correspond to the intended interpretation. Over a language that contains countably many symbols, we will have a countable term model, but some intended interpretations (e.g. of the language of set theory or of the language of calculus) are larger than countable. We will go into a little bit more detail on this later.

We can now prove the most important step for applying Henkin's method to predicate logic.

Theorem 5.3. *Let $\mathcal{L}$ be a language, Γ be a Henkin extension in $\mathcal{L}$ and M, I, f be the term model of Γ. Then for all $\mathcal{L}$-formulae A:*

$$M, I, f \models A \textit{ iff } A \in \Gamma.$$

We will use a *slightly* stronger induction technique for this (this helps with substitutions).

Definition 51 (Rank of a Formula)**.** Let $\mathcal{L}$ be a language. Define a function $\texttt{rk} : \texttt{wff}(\mathcal{L}) \to \mathbb{N}$ by recursion on the construction of the formulae.

- Base cases:
  - For all terms t_0 and t_1, $\texttt{rk}(t_0 \equiv t_1) = 0$.
  - For all n-ary relation symbols R and terms $t_0, ..., t_{n-1}$, $\texttt{rk}(Rt_0, ..., t_{n-1}) = 0$
  - $\texttt{rk}(\bot) = 0$.
- Recursive steps:
  - $\texttt{rk}(\neg B) = \texttt{rk}(B) + 1$.
  - $\texttt{rk}(B \vee C) = \max(\texttt{rk}(B), \texttt{rk}(C)) + 1$.
  - $\texttt{rk}(B \wedge C) = \max(\texttt{rk}(B), \texttt{rk}(C)) + 1$.
  - $\texttt{rk}(B \to C) = \max(\texttt{rk}(B), \texttt{rk}(C)) + 1$.
  - $\texttt{rk}(\forall x B) = \texttt{rk}(B) + 1$.
  - $\texttt{rk}(\exists x B) = \texttt{rk}(B) + 1$.

Intuitively, the rank of a formula is how ‘deep’ the longest embedding of operators goes. As the rank is just a natural number, we can do an induction on the rank.

Rank Induction

To prove that all formulae A numbers have property P, it suffices to show the following:

- **Base case:** Show that all formulae of rank 0 have property P.
- **Inductive step:** Assume that for a fixed but arbitrary n it is the case that all formulae of rank n or less have the property P (this is the **Induction Hypothesis**). Using this assumption, show that all formulae of rank $n + 1$ have the property P.

We can prove the effectiveness of Rank Induction either by our usual means of using minimal counterexample or we can observe that it is just a special case of Mathematical Induction on the natural numbers. We are now ready to prove the above theorem.

Proof. Let $\mathcal{L}$ be a language, Γ be a Henkin extension in $\mathcal{L}$ and M, I, f be the term model of Γ.

First prove by induction over the construction of terms that for all terms t, it is the case that $I^f(t) = [t]$

- Base 1. If $t = v$ is a variable symbol, then $I^f(t) = f(v) = [v]$ by definition of f.
- Base 2. If $t = c$ is a constant symbol, then $I^f(t) = I(c) = [c]$ by definition of I.
- Inductive step. Assume $t = Ft_0...t_{n-1}$ for an n-ary function symbol F and terms $t_0, ..., t_{n-1}$. Induction hypothesis: for all $i < n$, $I^f(t_i) = [t_i]$. Show that $I^f(t) = [t]$.

  By the definition of interpretation, $I^f(t) = I(F)(I^f(t_0), ..., I^f(t_{n-1}))$. By the induction hypothesis, this is equal to $I(F)([t_0], ..., [t_{n-1}])$. By the definition of I, this is equal to $[Ft_0...t_{n-1}]$, which is $[t]$. Thus $I^f(t) = [t]$, which was to show.

Now, show the following by Rank Induction on the formulae in $\mathcal{L}$.

For all A, it is the case that $M, I, f \models A$ iff $A \in \Gamma$.

To keep the proof short, we will only treat the base cases and the inductive steps for $\neg$, $\to$ and $\forall$ and tacitly assume we have abbreviated everything else.

- Base. Show for all formulae A of rank 0 that $M, I, f \models A$ iff $A \in \Gamma$. There are three cases.
  - Case 1. $A = t_0 \equiv t_1$ for terms t_0, t_1.

    If $M, I, f \models t_0 \equiv t_1$, then by definition of $\models$, it is the case that $I^f(t_0) = I^f(t_1)$, which by the above means that $[t_0] = [t_1]$. But this means that $t_0 \equiv t_1 \in \Gamma$.

    Conversely, if $t_0 \equiv t_1 \in \Gamma$, then $[t_0] = [t_1]$, so $I^f(t_0) = I^f(t_1)$, hence $M, I, f \models t_0 \equiv t_1$.
  - Case 2. $A = Rt_0 ... t_{n-1}$ for a an n-ary relation symbol R and terms $t_0, ..., t_{n-1}$.

    If $M, I, f \models Rt_0 ... t_{n-1}$, then $I(R)(I^f(t_0), ..., I^f(t_{n-1})) = 1$ by definition of $\models$. By the above, this means that $I(R)([t_0], ..., [t_{n-1}]) = 1$. By definition of I, this means that $Rt_0 ... t_{n-1} \in \Gamma$.

    Conversely, if $Rt_0 ... t_{n-1} \in \Gamma$, by definition of I, it is the case that $I(R)([t_0], ..., [t_{n-1}]) = 1$. By the above, then $I(R)(I^f(t_0), ..., I^f(t_{n-1})) = 1$ and by definition of $\models$ this means that $M, I, f \models A$.
  - Case 3. $A = \bot$. By definition of $\models$, it not the case that $M, I, f \models \bot$. And because Γ is consistent, $\bot \notin \Gamma$. Thus it is the case that $M, I, f \models \bot$ iff $\bot \in \Gamma$ because both are false.

  In all cases, we conclude what is to show.
- Inductive step. Let n be fixed but arbitrary. Induction hypothesis: for all formulae B of rank n or less it is the case that:$M, I, f \models B$ iff $B \in \Gamma$. Show that for all formulae A of rank $n + 1$, it is the case that $M, I, f \models A$ iff $A \in \Gamma$.

  We distinguish three cases: there is some B such that $A = \neg B$; there are B and C such that $A = B \to C$; and there are B and x such that $A = \forall x B$. Note that in all cases, the rank of B (and C) is n or less by definition of `rk`.
  - Case 1. $A = \neg B$ for some formula B. If $M, I, f \models \neg B$, then by definition of $\models$, it is not the case that $M, I, f \models B$. The rank of B is n or less, so by the contrapositive of the IH, $B \notin \Gamma$. Because Γ is maximally consistent, this means that B is inconsistent with Γ, i.e. $\Gamma \cup \{B\} \vdash \bot$. By ($\neg$I.), this means that $\Gamma \vdash \neg B$ and because Γ is deductively closed, hence $\neg B \in \Gamma$, i.e. $A \in \Gamma$.

    Conversely, if $\neg B \in \Gamma$ then because Γ is consistent, $B \notin \Gamma$ (otherwise we can derive $\bot$ from Γ by ($\neg$E)). The rank of B is n or less, so by the contrapositive of the IH, it is not the case that $M, I, f \models B$. By definition of $\models$, this means that $M, I, f \models \neg B$.
  - Case 2. $A = B \to C$ for some formulae B and C. Show that $M, I, f \models B \to C$ iff $B \to C \in \Gamma$.

    If $M, I, f \models B \to C$ then by definition of $\models$, either not $M, I, f \models B$ or $M, I, f \models C$.

  - Case 2.1. not $M, I, f \models B$. Then by the contrapositive of the IH, $B \notin \Gamma$. Because Γ is maximally consistent, $\Gamma \cup \{B\} \vdash \bot$, so by ($\neg$I.), $\Gamma \vdash \neg B$ hence because Γ is deductively closed, $\neg B \in \Gamma$. By (RAA) with an empty discharge it follows that $\Gamma \vdash B \to C$. Or, to give some detail the following is a proof from Γ.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\neg B \qquad [B]^1}{\bot}\,(\neg\text{E.}) \qquad [\neg C]^2}{\bot \wedge \neg C}\,(\wedge\text{I.})}{\bot}\,(\wedge\text{E.}_1)}{C}\,(\text{RAA})^2}{B \to C}\,(\to\text{I.})^1$$

    Thus $\Gamma \vdash B \to C$, hence because Γ is deductively closed, $B \to C \in \Gamma$.

  - Case 2.2. $M, I, f \models C$. Then by the IH, $C \in \Gamma$. Then the following is a proof from Γ.

$$\dfrac{\dfrac{\dfrac{C \qquad [B]^1}{C \wedge B}\,(\wedge\text{I.})}{C}\,(\wedge\text{E.}_1)}{B \to C}\,(\to\text{I.})^1$$

    Thus $\Gamma \vdash B \to C$, hence because Γ is deductively closed, $B \to C \in \Gamma$.

  In both cases, $B \to C \in \Gamma$, which was to show.

  Conversely, assume $B \to C \in \Gamma$ and show that $M, I, f \models B \to C$. Towards a *reductio*, assume that not $M, I, f \models B \to C$. By defintion of $\models$, this means that $M, I, f \models B$ and not $M, I, f \models C$. By the IH for B and the contrapositive of the IH for C, then $B \in \Gamma$ and $C \notin \Gamma$. But From $B \to C \in \Gamma$ and $B \in \Gamma$ it follows by ($\to$E.) that $\Gamma \vdash C$. As Γ is deductively closed, $C \in \Gamma$, contradiction to $C \notin \Gamma$. Thus by *reductio*, $M, I, f \models B \to C$.

- Case 3. $A = \forall x B$ for a formula B and a variable x. Show that $M, I, f \models \forall x B$ iff $\forall x B \in \Gamma$. Show the contraposition of this, i.e. not $M, I, f \models \forall x B$ iff not $\forall x B \in \Gamma$

  Left-to-right. Assume not $M, I, f \models \forall x B$. Then, by definition of $\models$, there is an assignment function g that differs from f at most in what it assigns to x such that not $M, I, g \models B$. We know that $g(x) \in M$, so there is a term t such that $g(x) = [t]$, which means that $g(x) = I^f(t)$. By the Substitution Lemma, not $M, I, f \models B[t/x]$. By the IH, $B[t/x] \notin \Gamma$. Then, assume for *reductio* that $\forall x B \in \Gamma$. By ($\forall$E), this means that $\Gamma \vdash B[t/x]$, which contradicts that $B[t/x] \notin \Gamma$. Thus, by *reductio*, $\forall x B \notin \Gamma$. This was to show.

  Right-to-left. If not $\forall x B \in \Gamma$, then $\Gamma \cup \{\forall x B\} \vdash \bot$ as Γ is maximally consistent. By the usual argument (cf. the step for negation), $\Gamma \vdash \neg \forall x B$, so as Γ is deductively closed $\neg \forall x B \in \Gamma$. As Γ contains witnesses, this means that $\neg B[c_B/x] \in \Gamma$. Thus, $B[c_B/x] \notin \Gamma$ (as otherwise Γ would be inconsistent). The formula $B[c_B/x]$ has lower complexity than A, so by the IH, it is not the case that $M, I, f \models B[c_B/x]$. Define an assignment g by $g(x) = I^f(c_B)$ and $g(v) = f(v)$ for all variables $v \neq x$. By the contrapositive of the Substitution Lemma, not $M, I, g \models B$. Because g differs from

f at most in what it assigns to x, by definition of $\models$, it follows that not $M, I, g \models \forall x B$, which was to show.

Remark: The uses of the Substitution Lemma here are why we use Rank Induction. $B[t/x]$ may be different from B so with an induction on construction, we could not use the IH; because t may be very long, we also cannot use induction on the length of a formula. But the *rank* of $B[t/x]$ is the same as the rank of B (and thus less than the rank of A) regardless of what t is.

We have covered all cases and concluded in each what was to show.

This concludes the induction. □

We can immediately conclude the following.

Theorem 5.4. *Let $\mathcal{L}$ be a language and Γ be a Henkin extension in $\mathcal{L}$. Then Γ is satisfiable.*

Proof. Let $\mathcal{L}$ be a language and Γ be a Henkin extension in $\mathcal{L}$. Let M, I, f be the term model for Γ. By the previous theorem, for all $A \in \Gamma$, it is the case that $M, I, f \models A$. Thus M, I, f satisfies Γ. □

Now to prove completeness, we need to find a way to extend each consistent set Γ to a Henkin extension $\hat{\Gamma}$ with $\hat{\Gamma} \supseteq \Gamma$.

5.2 Finding Henkin Extensions

We will use the following result without proof.

Fact 5.5. *Let $\mathcal{L}$ be a language and Γ be a consistent set of formulae. Then there is a maximally consistent set Γ' such that $\Gamma \subseteq \Gamma'$.*

For countable languages $\mathcal{L}$, $\texttt{wff}(\mathcal{L})$ is countable, so for such languages we can do this as in propositional logic. However, this is not in general the case anymore (e.g. we could have a language with more than countably many constant symbols and then get more than countably many wffs). In the propositional case, we used the Axiom of Countable Choice to enumerate the formulae and then stepwise decide whether the n-th formulae can be added consistently to Γ. Here, we can do the same thing, but we need the full Axiom of Choice and an advanced induction method called *transfinite induction*. Equivalently, we may use Zorn's Lemma (which is equivalent to the Axiom of Choice and proven from it by transfinite induction).

Then we can start constructing a Henkin extension by the following recursion on the natural numbers. Fix some language $\mathcal{L}$ and a consistent set Γ of $\mathcal{L}$-formulae. We define for each natural number n a language $\mathcal{L}_n = \langle \mathcal{C}_n, \mathcal{F}_n, \mathcal{R}_n \rangle$ and a set Γ_n of formulae in $\mathcal{L}_n$.

- Base case. $\mathcal{L}_0 = \mathcal{L}$ and $\Gamma_0 = \Gamma$.
- Recursive step. Assume we have defined $\mathcal{L}_n$ and Γ_n.

For each $A \in \mathcal{L}_n$ let c_A be a symbol not used in $\mathcal{L}_n$ and define $\mathcal{L}_{n+1} = \langle \mathcal{C}_{n+1}, \mathcal{F}_{n+1}, \mathcal{R}_{n+1} \rangle$ as follows:

$$\mathcal{C}_{n+1} = \mathcal{C} \cup \{c_A \mid A \in \mathtt{wff}(\mathcal{L}_n)\}, \mathcal{F}_{n+1} = \mathcal{F}_n, \text{ and } \mathcal{R}_{n+1} = \mathcal{R}_n.$$

Let $\Gamma_{n+1} \subseteq \mathtt{wff}(\mathcal{L}_{n+1})$ be a maximally consistent superset of the following set:

$$\Gamma_n \cup \{\neg A[c_A/x] \mid A \in \mathtt{wff}(\mathcal{L}_n) \text{ and } x \in \mathcal{V} \text{ and } \neg\forall x A \in \Gamma_n\}.$$

It is not obvious that we can define such a recursion. To be able to just 'let' Γ_{n+1} be as desired, we need to prove for every n that such a maximally consistent Γ_{n+1} exist. We do so by proving that that for all n, $\Gamma_n \cup \{\neg A[c_A/x] \mid A \in \mathtt{wff}(\mathcal{L}_n)$ and $x \in \mathcal{V}$ and $\neg\forall x A \in \Gamma_n\}$ is consistent and then applying the Fact.

The proof will be sketched, not working out in detail some proof trees and using an 'iteration' where formally there would be an induction.

First, we need a small lemma.

Lemma 5.6. *Let Γ be a set of formulae and A be a formula. Let c be a constant not occurring in Γ or A and x be a variable not occurring anywhere in Γ. If $\Gamma \vdash A[c/x]$, then $\Gamma \vdash \forall x A$.*

Proof. If $\Gamma \vdash A[c/x]$ and c does not occur in Γ, then we can replace everywhere in the proof c by x and the proof is still valid and we haven't changed Γ. (Formally, we would do this by induction on the length of proofs, but it is obvious enough.)

Then we have shown that $\Gamma \vdash A$ and as by assumption x does not occur free in Γ, it follows by (∀I.) that $\Gamma \vdash \forall x A$. □

Theorem 5.7. *Suppose Γ_n is a consistent set of formulae in the language $\mathcal{L}_n$. Then the following is a consistent set of formulae in the language $\mathcal{L}_{n+1}$.*

$$\Gamma_n \cup \{\neg A[c_A/x] \mid A \in \mathtt{wff}(\mathcal{L}_n) \textit{ and } x \in \mathcal{V} \textit{ and } \neg\forall x A \in \Gamma_n\}.$$

Proof. Show the contrapositive of the theorem.

Assume that $\Gamma_n \cup \{\neg A[c_A/x] \mid A \in \mathtt{wff}(\mathcal{L}_n)$ and $x \in \mathcal{V}$ and $\neg\forall x A \in \Gamma_n\} \vdash \bot$.

Since proofs are finite, there is a finite $\Gamma' \subseteq \Gamma_n$ and finitely many variables $x_0, ..., x_{n-1}$ and $\mathcal{L}_n$-formulas $A_0, ..., A_{n-1}$ such that for all $i < n$, $\neg\forall x_i A_i \in \Gamma_n$ and $\Gamma' \cup \{\neg A_i[c_{A_i}/x_i] \mid i < n\} \vdash \bot$.

We may assume that for all $i < n$, $\neg\forall x A_i \in \Gamma'$, as adding finitely many formulae to Γ' does not change anything.

Then proceed as follows.

- By (RAA), $\Gamma' \cup \{\neg A_i[c_{A_i}/x_i] \mid i < n-1\} \vdash A_{n-1}[c_{A_{n-1}}/x_{n-1}]$.

- Because only finitely many variables can occur throughout this proof, we can pick an unused variable y and let $B = A_{n-1}[y/x_{n-1}]$.
- It follows that $\Gamma' \cup \{\neg A_i[c_{A_i}/x_i] \mid i < n-1\} \vdash B[c_{A_{n-1}}/y]$
- By the lemma, $\Gamma' \cup \{\neg A_i[c_{A_i}/x_i] \mid i < n-1\} \vdash \forall y B$.
- We can now 'rename' y to x_{n-1} in $\forall y B$ to obtain $\Gamma' \cup \{\neg A_i[c_{A_i}/x_i] \mid i < n-1\} \vdash \forall x_{n-1} A_{n-1}$.

  'Renaming' goes as follows: $\forall y B \vdash B[y/y]$ by ($\forall$E.); but $B[y/y] = B$ by the definition of Substitution and $B = A_{n-1}[y/x_{n-1}]$ by definition of B. Thus $\forall y B \vdash A_{n-1}[y/x_{n-1}]$. But because y does not occur in A_{n-1} or premisses, by ($\forall$I.), it follows that $\forall y B \vdash \forall x_{n-1} A_{n-1}$.

  (**Remark**: the previous Lemma provides that if we have an unused constant symbol, we get to introduce a universal quantifier. But we cannot guarantee that x_{n-1} does not occur in premisses, so to use the Lemma, we must replace x_{n-1} with an unused y, apply the Lemma, and then replace y with x_{n-1} again. This is just some rewriting of formulae to make things the way we need them to be and there is no good 'intuitive' description of what is going on beyond this.)
- But $\neg\forall x_{n-1} A_{n-1} \in \Gamma'$, thus $\Gamma' \cup \{\neg A_i[c_{A_i}/x_i] \mid i < n-1\} \vdash \bot$ by ($\neg$E.).

Iterate this process n times (formally, do an induction), each time getting rid of one premiss. Conclude:

- $\Gamma' \vdash \bot$.

Thus $\Gamma_n \vdash \bot$, since $\Gamma' \subseteq \Gamma_n$. This was to show. □

The theorem ensures that the induction to define $\mathcal{L}_n$ and Γ_n works.

Now define $\hat{\Gamma} = \bigcup_{n\in\mathbb{N}} \Gamma_n$ and $\hat{\mathcal{L}} = \langle \hat{\mathcal{C}}, \hat{\mathcal{F}}, \hat{\mathcal{R}} \rangle$ by $\hat{\mathcal{C}} = \bigcup_{n\in\mathbb{N}} \mathcal{C}_n$, $\hat{\mathcal{F}} = \mathcal{F}_0$ and $\hat{\mathcal{R}} = \mathcal{R}_0$.

Theorem 5.8. *$\hat{\Gamma}$ is maximally consistent and contains witnesses (i.e. a Henkin extension).*

Proof. First show that $\hat{\Gamma}$ is consistent. Towards a *reductio* assume it is not. Then, because proofs are finite, there is a finite subset $\Gamma' \subseteq \hat{\Gamma}$ such that $\Gamma' \vdash \bot$. For each $A \in \Gamma'$ let n_A be the smallest n such that $A \in \Gamma_n$. Let n be the maximum of $\{n_A \mid A \in \Gamma'\}$. Then $\Gamma' \subseteq \Gamma_n$. But this means that $\Gamma_n \vdash \bot$ so Γ_n is inconsistent, which contradicts the previous result. By *reductio*, $\hat{\Gamma}$ is consistent.

Second show that $\hat{\Gamma}$ is maximally consistent. Towards a *reductio* assume that it is not. Then there is some $\hat{\mathcal{L}}$-formula $A \notin \hat{\Gamma}$ such that $\hat{\Gamma} \cup \{A\}$ is consistent. For each constant symbol c in A let n_c be the smallest n such that $c \in \mathcal{C}_n$. Let n be the maximum of $\{n_c \mid c \text{ occurs in } A\}$. Then A is a formula in $\mathcal{L}_n$. Because $A \notin \hat{\Gamma}$ it must be the case that $A \notin \Gamma_n$. But Γ_n is maximally consistent, so $\Gamma_n \cup \{A\} \vdash \bot$. Hence also $\hat{\Gamma} \cup \{A\} \vdash \bot$. Contradiction to $\hat{\Gamma}$ being consistent. By *reductio*, $\hat{\Gamma}$ is maximally consistent.

Third show that $\hat{\Gamma}$ contains witnesses. Let x be an arbitrary variable and A be an arbitrary $\hat{\mathcal{L}}$-formula such that $\neg\forall x A \in \hat{\Gamma}$. Because $\neg\forall x A \in \hat{\Gamma}$ and $\hat{\Gamma} = \bigcup_{n\in\mathbb{N}} \Gamma_n$, there is some n such that $\neg\forall x A \in \Gamma_n$. By construction of Γ_{n+1}, there is a $c_A \in \mathcal{C}_{n+1}$ such that $\neg A[c_A/x] \in \Gamma_{n+1}$. Since $\Gamma_{n+1} \subseteq \hat{\Gamma}$, it follows that $\neg A[c_A/x] \in \hat{\Gamma}$. As x and A were arbitrary, it follows that for all x and A, if $\neg\forall x A \in \hat{\Gamma}$, there is a constant symbol c_A such that $\neg A[c_A/x] \in \hat{\Gamma}$. Thus $\hat{\Gamma}$ contains witnesses. □

We call $\hat{\Gamma}$ the **Henkin extension of** Γ. Now it only remains to put everything together.

5.3 Proof of Completeness

Theorem 5.9 (Henkin's Model Existence Theorem)**.** *Let $\mathcal{L}$ be a language and Γ be a consistent set of $\mathcal{L}$-formulae. Then there is a model M, I, f such that $M, I, f \models B$ for all $B \in \Gamma$.*

Proof. Let $\mathcal{L} = \langle \mathcal{C}, \mathcal{F}, \mathcal{R}, \alpha \rangle$ be a language and Γ be a consistent set of $\mathcal{L}$-formulae. By the construction and results of the previous section, there is a language $\hat{\mathcal{L}}$ and a set $\hat{\Gamma}$ of $\hat{\mathcal{L}}$-formulae such that $\Gamma \subseteq \hat{\Gamma}$ and $\hat{\Gamma}$ is a Henkin extension. Let M, I, f be the term model of $\hat{\Gamma}$. By Theorem 5.3, $M, I, f \models B$ for all $B \in \hat{\Gamma}$.

Now define a $\mathcal{L}$-model M', I', f' as follows.

- $M' = M$;
- for all constant symbols $c \in \mathcal{C}$, $I'(c) = I(c)$;
- for all function symbols $F \in \mathcal{F}$, $I'(F) = I(F)$;
- for all relation symbols $R \in \mathcal{R}$, $I'(R) = I(R)$;
- $f' = f$.

Now let $A \in \Gamma$. Because also $A \in \hat{\Gamma}$, $M, I, f \models A$. But since by definition of $\models$, this only depends on how I interprets the symbols in $\mathcal{L}$, $M', I', f' \models A$. (To make this entirely proper, this could be shown by a simple induction on the construction of A.) As this goes for all $A \in \Gamma$, Γ is satisfiable. □

Completeness follows immediately.

Proof of Completeness from Model Existence. Towards a *reductio* assume that there is a set Γ and formula A such that $\Gamma \models A$ and $\Gamma \not\vdash A$.

If $\Gamma \not\vdash A$, then $\Gamma \cup \{\neg A\}$ is consistent (as if $\Gamma \cup \{\neg A\}$ is inconsistent, then $\Gamma \vdash A$ by (RAA)). Thus by the Model Existence Theorem, there is a model M, I, f such that $M, I, f \models B$ for all $B \in \Gamma$ and $M, I, f \models \neg A$. The latter means that not $M, I, f \models A$ by definition of $\models$. But by definition of semantic consequence, this means it is not the case that $\Gamma \models A$. Contradiction to our assumption. Thus by *reductio*, $\Gamma \vdash A$. □

Note however that the proof of completeness is *non-constructive*. It tells you that *there is* a proof, but not what the proof is. The chances are slim of being able to extract an explicit proof of A from $\Gamma \models A$ and Henkin's construction. Note how we have used multiple language extensions and some non-constructive techniques with the Axiom of Choice that we cannot 'unroll'.

However, we are not in general interested in extracting such proofs. As already mentioned, finding a proof is typically easier than checking truth in all models—and now we don't even have a coinciding valuations theorem to help us with the latter. Thus we can rest assured that if we go the (comparatively) 'easy' way of finding proofs, we are not missing any semantic consequences. Besides this practical benefit, the Completeness theorem entails the Compactness theorem, which is extremely useful in its own right.

6 Compactness and the Löwenheim-Skolem Theorems

6.1 First Order Theories

From the perspective of our meta-language, we can look at an object language $\mathcal{L}$ and speak of its intended interpretation. But it would be desirable to fix the meaning of the object language from within it, i.e. we would like to specify the meaning of, say, the function symbol '$+$', by writing down an object language sentence involving '$+$' so that whenever the sentence is true, '$+$' has the right meaning. We can (at least approximate) the meaning of a language in *theories*.

Definition 52. Let $\mathcal{L}$ be a language. A $\mathcal{L}$-formula A is called **closed** if it has no free variables (closed formulae are also called **sentences**). A set of closed formulae in $\mathcal{L}$ is a **theory** in $\mathcal{L}$.

For example, the set of Robinson Axioms is a theory. It is called *Robinson Arithmetic* and is typically denoted by the letter **Q**. Robinson Arithmetic gets many facts about the meaning of the language of arithmetic right, but also leaves many aspects underspecified. For example, recall the Robinson axioms for addition.

$\forall x(x + 0 \equiv x)$ and $\forall x \forall y(x + sy \equiv s(x + y))$.

This already goes *somewhere*. For example, for all natural numbers n and m, $s^n0 + s^m0 \equiv s^m0 + s^n0$ is true in every model of **Q**. So, it is tempting to conclude, **Q** expresses in the object language that addition is commutative. But this is not quite right, as there are models of **Q** in which it is not the case that $\forall x \forall y(+xy \equiv +yx)$. The following structure M, I is an example.

- $M = \mathbb{N} \cup \{\heartsuit, \clubsuit\}$.
- $I(0) = 0$.
- $I(s)(n) = n + 1$, $I(s)(\heartsuit) = \clubsuit$ and $I(s)(\clubsuit) = \heartsuit$.
- $I(+)$ is defined by the following table. (where n and m are members of $\mathbb{N}$).

$I(+)$	n	$\heartsuit$	$\clubsuit$
m	$m+n$	$\clubsuit$	$\heartsuit$
$\heartsuit$	$\heartsuit$	$\clubsuit$	$\heartsuit$
$\clubsuit$	$\clubsuit$	$\clubsuit$	$\heartsuit$

  That is, for actual numbers n, m, $I(+)$ is as the intended interpretation, but for the 'deviant' elements $\heartsuit$ and $\clubsuit$, we interpret $+$ such that is the case that $I(+)(m, \heartsuit) = \clubsuit$, $I(+)(m, \clubsuit) = \heartsuit$; and $I(+)(\heartsuit, n) = \heartsuit$, $I(+)(\heartsuit, \heartsuit) = \clubsuit$ *etc.*

- See below for an interpretation of $\cdot$.

Let f be any assignment. It is easy to check that $M, I, f \models (\forall x(x + 0 \equiv x)) \wedge (\forall x \forall y(x + sy \equiv s(x + y)))$ and also satisfies the other Robinson axioms. But note that for any number n, $I(+)(n, \heartsuit) = \clubsuit$, but $I(+)(\heartsuit, n) = \heartsuit$. So it is not the case that $M, I, f \models \forall x \forall y(+xy \equiv +yx)$.

A similarly 'deviant' interpretation of multiplication that is as follows.

$I(\cdot)$	0	$n > 0$	$\heartsuit$	$\clubsuit$
0	0	0	$\heartsuit$	$\clubsuit$
$m > 0$	0	$m \cdot n$	$\heartsuit$	$\clubsuit$
$\heartsuit$	0	$\clubsuit$	$\clubsuit$	$\clubsuit$
$\clubsuit$	0	$\heartsuit$	$\heartsuit$	$\heartsuit$

A theory that fixes this problem is *Peano Arithmetic*. This theory improves on Robinson Arithmetic by adding one more thing we know about the natural numbers: that we can prove things about them by induction.

Definition 53. The theory **PA** in the language of arithmetic is the following (infinite) set of closed formulae.

1. $\forall x(\neg sx \equiv 0)$.
2. $\forall x \forall y(sx \equiv sy \to x \equiv y)$.
3. $\forall x(x + 0 \equiv x)$ and $\forall x \forall y(x + sy \equiv s(x + y))$.
4. $\forall x(x \cdot 0 \equiv 0)$ and $\forall x \forall y(x \cdot sy \equiv (x \cdot y) + x)$.
5. For all n and all formulae $A(x_1, ..., x_n, y)$ in $n + 1$ free variables:

   $\forall x_1 ... \forall x_n((A[y/0] \wedge \forall y(A \to A[sy/y])) \to \forall y A)$.

The entry in (5) is called the *induction scheme*. It is the object language version of our meta-language proof method of Mathematical Induction. You can prove by Mathematical Induction that for all numbers m and n, it is the case that **Q** $\models s^n 0 + s^m 0 \wedge s^m 0 + s^n 0$. Using the induction scheme of **PA**, you can formalise this argument and show that **PA** $\models \forall x \forall y(+xy \equiv +yx)$.

Attention: Assume you have shown that for some formula A and all numbers n, **PA** $\models A[s^n 0/x]$. This does **not** mean that **PA** $\models \forall x A$. It depends on *how* you prove the statement about $s^n 0$ (by induction or otherwise) and whether this can be formalised in the object language.

Does **PA** specify the meaning of the language of arithmetic? We can make more precise what could be the 'best' outcome here. We know the intended interpretation $\mathbb{N}, I$, so how about we characterise the meaning of language of arithmetic as all sentences true in this model.

Definition 54. Let $\mathcal{M}$ be a structure. Fix any arbitrary assignment f. The **theory of** $\mathcal{M}$ is the set $\mathrm{Th}(\mathcal{M})$ of all closed formulas A such that $\mathcal{M}, f \models A$.

That is, $\mathrm{Th}(\mathcal{M}) = \{A \mid \texttt{frvar}(A) = \emptyset \text{ and } \mathcal{M}, f \models A\}$.

Note that for closed formulae the assignment does not matter, so the choice of f does not matter.

The theory of a model has an important property. It is *complete*.

Definition 55. A theory is **complete** iff for all closed formulae A, either $T \vdash A$ or $T \vdash \neg A$.

Attention: This is not the completeness of the Completeness Theorem. The completeness theorem demonstrates the completeness that $\vdash$ has with respect to $\models$. The completeness of this definition is a property of a set of sentences, not of a consequence relation.

Note that if $\mathcal{M}$ is a structure and f is an assignment, then for all A, either $\mathcal{M}, f \models A$ or $\mathcal{M}, f \models \neg A$. Thus the theory of a model is complete.

The theory of $\mathbb{N}, I$ is called *true arithmetic* and denoted by **TA**. Certainly, **TA** fixes the intended interpretation of arithmetic in the object language as well as is possible. But not in a very helpful way: do you know what the members of **TA** are? Only in a tautological way: you know that the members of **TA** are the truths in the intended interpretation. But this is not very helpful. In contrast, you know the members of **Q** and **PA** in a very concrete sense that does not depend on your use of the intended interpretation.

6.2 The Compactness Theorem

As usual, we can get the following from the completeness theorem.

Theorem 6.1. *Let $\mathcal{L}$ be a language, Γ be a set of $\mathcal{L}$-formulae and A be a $\mathcal{L}$-formula. If $\Gamma \models A$, then there is a finite $\Gamma' \subseteq \Gamma$ such that $\Gamma' \models A$.*

Proof. Let $\mathcal{L}$, Γ and A as in the theorem. Suppose $\Gamma \models A$. Then by Completeness, $\Gamma \vdash A$. As proofs are finite, we can let Γ' be the finite set of premisses required for the proof of A. Then $\Gamma' \vdash A$. By Soundness, $\Gamma' \models A$. □

The following theorem is what is typically called the **compactness theorem**. It follows immediately from the fact that $\models$ is compact.

Theorem 6.2 (Compactness Theorem)**.** *Let $\mathcal{L}$ be a language and Γ be a set of $\mathcal{L}$-formulae. Then: Γ is satisfiable iff all finite subsets of Γ are satisfiable.*

Proof. Left-to-right is trivial: if Γ has a model M, I, f then M, I, f is also a model of any subset.

For right-to-left, suppose that all finite subsets of Γ are satisfiable. Towards a *reductio* that Γ is not satisfiable. This means that $\Gamma \models \bot$, since if there is *no* model M, I, f such that $M, I, f \models B$ for all $B \in \Gamma$, then it is vacuously the case that all M, I, f that make all members of Γ true, also make $\bot$ true. By the compactness of $\models$, there is a finite Γ' such that $\Gamma' \models \bot$.

But $\Gamma' \models \bot$ means that Γ' is not satisfiable. This is because by definition of $\models$, $\Gamma' \models \bot$ means that any model M, I, f such that $M, I, f \models B$ for all $B \in \Gamma'$ it is the case that $M, I, f \models \bot$. But the latter cannot be the case by definition of $\models$, so there is no such model. This contradicts the assumption that all finite subsets of Γ' are satisfiable. By *reductio*, Γ is satisfiable. □

The compactness theorem has *shocking* consequences. Here is one of them.

Theorem 6.3. *There is a structure M, I with an $a \in M$ such that for any assignment f with $f(x) = a$ and for all $n \in \mathbb{N}$, it is the case that $M, I, f \models \neg x \equiv s^n 0$ and M, I, f is a model of* $\mathbf{TA}$.

That is, there are models of true arithmetic that have elements that do not correspond to an actual natural number (recall that the actual natural numbers are obtained by starting with 0 and recursively taking successors.) Such models as are provided by the theorem are called *non-standard models of arithmetic* and elements like a are called *non-standard numbers.*

Proof. Let $\mathcal{L}^A$ be the language of arithmetic and let $\mathcal{L}$ be the language obtained by adding a new constant symbol c to the language. Then consider the following set of formulae in $\mathcal{L}$.

$$\Gamma = \mathbf{TA} \cup \{\neg c \equiv s^n 0 \mid n \in \mathbb{N}\}.$$

Show that Γ is satisfiable. By the Compactness Theorem, it suffices to show that every finite $\Gamma' \subseteq \Gamma$ is satisfiable. So let $\Gamma' \subseteq \Gamma$ be finite and otherwise arbitrary.

As Γ' is finite, there is some natural number k such that for all $n \geq k$, $\neg c \equiv s^n 0 \notin \Gamma'$. Then let $\mathbb{N}, I$ be the standard model of arithmetic and extend it to a model in $\mathcal{L}$ by interpreting c as $I(c) = k$. Let f be any assignment function.

Then $\mathbb{N}, I, f \models B$ for all $B \in \Gamma'$. This is because if $B \in \Gamma'$, then either $B \in \mathbf{TA}$ or $B = \neg c \equiv s^n 0$ for some $n < k$.

- Case 1. $B \in \mathbf{TA}$. Then $\mathbb{N}, I, f \models B$ by definition of $\mathbf{TA}$.
- Case 2. $B = \neg c \equiv s^n 0$ for some $n < k$. Then $I^f(s^n 0) = n$ and $I^f(c) = k$ by definition of I and as $n < k$, $I^f(s^n 0) \neq I^f(c)$, so $\mathbb{N}, I, f \models \neg c \equiv s^n 0$ by definition of $\models$.

So $\mathbb{N}, I, f$ is a model of Γ'. As this goes for arbitrary Γ', all finite subsets of Γ are satisfiable, so Γ is satisfiable. Let M', I', f' be a model of Γ. Let $a = I'(c)$. Clearly, $a \neq I^f(s^n 0)$ for all n, as otherwise there is an n such that $M', I', f' \models c \equiv s^n 0$.

Now define $M = M'$ and I to be like I' except that it does not assign anything to c. M, I is a structure in $\mathcal{L}^A$. Let f be any assignment with $f(x) = a$. Then $M, I, f \models \neg x \equiv s^n 0$ for all $n \in \mathbb{N}$ since $I^f(s^n 0) = I'^{f'}(s^n 0) \neq a$. □

The result might be less shocking once you realise that you cannot express 'x is a nonstandard number' in the object language. We use our meta-language understanding of the actual natural numbers to 'see' that the standard numbers are formed as $s^n 0$ for $n \in \mathbb{N}$, but since we do not have access to $\mathbb{N}$ from the object language, we cannot do this.

As a matter of fact, the previous theorem *shows that* there is no formula $A(x)$ in the language of arithmetic that expresses 'x is a nonstandard number'.

Proof. Towards a *reductio* assume there is such an $A(x)$. Then $\exists x A$ is a closed formula in the language of arithmetic. We know that the standard model $\mathbb{N}, I$ does not contain nonstandard numbers, thus $\neg \exists x A \in$

TA. But the previous theorem provides a model M, I, f of **TA** with $M, I, f \models \exists x A$. Thus $M, I, f \models (\neg \exists x A) \land (\exists x A)$. This can't be. Contradiction. By *reductio*, there is no such A. □

Thus although **TA** fixes the intended meaning of the symbols in the language of arithmetic, it fails to fix the true natural numbers. This is a first result about the *expressive limitations of first order logic.* Some properties of models cannot be fixed by a theory, not even by a complete theory like **TA**.

We will now see more radical failures of this kind.

6.3 Sizes of Sets

We saw earlier that some sets are *countable* in that they can be enumerated by the natural numbers. Some sets are larger than that. Let us introduce some simple notation.

Definition 56. Given a set s, let $|s|$ be the **size** of s.

The size of a set is also called its **cardinality**, but this term implies theory we have not developed here. Instead of saying what sizes *are*, we will just say something about how they compare.

Definition 57. Let s and t be sets. A function $f : s \to t$ is called **one-to-one** (or **bijective**) iff$_{\text{Def}}$ for all $y \in t$ there is a unique $x \in s$ such that $f(x) = y$.

A one-to-one function pairs up all elements of s and t such that everyone has a partner. We now define two sets to have the same size if there is a one-to-one function between them.

Definition 58. Let s and t be sets.

- $|s| = |t|$, iff$_{\text{Def}}$ there is a one-to-one function $f : s \to t$.
- $|s| \leq |t|$, iff$_{\text{Def}}$ there is a subset $t' \subseteq t$ such that $|s| = |t'|$.
- $|s| < |t|$, iff$_{\text{Def}}$ $|s| \leq |t|$ and not $|s| = |t|$.

Intuitively, this should stand to reason: if we take two sets s and t and we can pair up every member of s with a member in t without leaving out any member of either s or t then there are 'equally many' elements in both sets.

Consider how you would ensure that you have equally many of two kinds if you would not know numbers. Say, you have a heap of forks and knives and you want to ensure that you have as many forks as you have knives. Since you don't know numbers, you can't just count the forks and the knives. But you *can* pick a fork, find a knife, put the pair to the side and keep going until either (a) you have a fork left over, but no knife to pair it with, (b) a knife left over, but no fork to pair it with or (c) neither forks nor knives left over. In (a) you know you have more forks than knives; in (b) more knives than forks; and in (c) equally many forks and knives.

In the current situation, you indeed do not know the numbers to count the members of sets, since we have not defined what kind of object a *size* is. So using one-to-one pairings is your only option. That's what the definition above does.

Example. Some examples.

- All infinite countable sets are the same size. Let s and t be infinite countable, then we can write $s = \{s_i \mid i \in \mathbb{N}\}$ and $t = \{t_i \mid i \in \mathbb{N}\}$ without multiply-mentioning members. Then we can define a one-to-one function $f : s \to t$ as $f(s_i) = t_i$ for all $i \in \mathbb{N}$.
- In particular, the set of even numbers has the same size as the set of all numbers. As do the odd numbers, prime numbers, square numbers etc.
- All finite sets are properly smaller in size than the natural numbers, i.e. for all finite s, $|s| < |\mathbb{N}|$. If s contains n elements, then we straightforwardly find a one-to-one mapping from s to the set $\{0, ..., n-1\} \subseteq \mathbb{N}$. So $|s| \leq |\mathbb{N}|$.

  Clearly it cannot be the case that $|s| = |\mathbb{N}|$. Let $f : s \to \mathbb{N}$ be any function. Then $\{f(x) \mid x \in s\}$ is a finite subset of $\mathbb{N}$, so it has a maximum n. Then there is no $x \in s$ such that $f(x) = n + 1$, so f is not one-to-one.

  Thus $|s| < |\mathbb{N}|$.

We now show that there are also *larger* sets than $\mathbb{N}$.

Definition 59 (Power Set)**.** Let s be a set. The **power set** of s, $\mathcal{P}(s)$ is the set of all subsets of s. Formally: $\mathcal{P}(s) = \{t \mid t \subseteq s\}$.

The following is another seminal result in Set Theory and the foundations of mathematics.

Theorem 6.4 (Cantor's Theorem)**.** *For all s, $|s| < |\mathcal{P}(s)|$.*

Proof. Clearly $|s| \leq |\mathcal{P}(s)|$, since the function $f : s \to \mathcal{P}(s)$ defined by $f(x) = \{x\}$ is one to one. It remains to show that it is not the case that $|s| = |\mathcal{P}(s)|$. Towards a *reductio* assume $|s| = |\mathcal{P}(s)|$, i.e. that there is a one-to-one function $f : s \to \mathcal{P}(s)$.

Consider the set $t = \{x \in s \mid x \notin f(x)\}$. Clearly $t \subseteq s$, so $t \in \mathcal{P}(s)$. Because f is one-to-one, there is a $y \in s$ such that $f(y) = t$.

Now, by the Law of Excluded Middle (in the meta-language), either $y \in t$ or $y \notin t$.

- Case 1. $y \in t$. By definition of t this means that $y \notin f(y)$. But $f(y) = t$, so $y \notin t$. Contradiction.
- Case 2. $y \notin t$. By definition of t this means that $y \in f(y)$. But $f(y) = t$, so $y \in t$. Contradiction.

In either case we reach a contradiction. Thus by *reductio*, $|s| \neq |\mathcal{P}(s)|$. This was to show. □

Thus, in particular, the set $\mathcal{P}(\mathbb{N})$ is larger than countable—it is *uncountable*. By iteratively taking power sets, Cantor's theorem delivers that there are infinitely many possible sizes of infinite sets. Once you have found countably many infinites, you can take their union and then keep going. There is an uncountably large infinity of ever-larger infinites.

6.4 Sizes of Models

Back to our models of first-order logic. As an immediate consequence of Henkin's Model Existence Theorem, we get the following.

Definition 60 (Size of Languages)**.** Let $\mathcal{L} = \langle \mathcal{C}, \mathcal{F}, \mathcal{R}, \alpha \rangle$ be a language. The **size** of $\mathcal{L}$ is $|\mathcal{C} \cup \mathcal{F} \cup \mathcal{R}|$.

Theorem 6.5 (Countable Model Theorem)**.** *Let $\mathcal{L}$ be a countable language. Let Γ be a set of $\mathcal{L}$-sentences that has an infinite model. Then Γ has a countable model.*

The Countable Model Theorem is sometimes known as Löwenheim's Theorem or the Löwenheim-Skolem Theorem (although we will see below a different theorem of that name). It is, arguably, the historically first non-trivial result in what is now known as Model Theory.

We will only give an extended sketch of the argument.

Proof. Let $\mathcal{L} = \langle \mathcal{C}, \mathcal{F}, \mathcal{R}, \alpha \rangle$ and Γ be as in the theorem. Let $\{c_i \mid i \in \mathbb{N}\}$ be a countably infinite set of unused constant symbols and extend $\mathcal{L}$ to the language $\mathcal{L}^*$ containing them them. Consider the following set Γ^* in the extended language.

$$\Gamma^* = \Gamma \cup \{\neg c_i \equiv c_j \mid i \in \mathbb{N}, j \in \mathbb{N} \text{ and } i \neq j\}.$$

That is, Γ^* includes all members of Γ and says that all of the new constant symbols have to receive non-identical interpretations in a model. It is easy to see that because Γ has an infinite model, Γ^* is satisfiable (just take the model and interpret all c_i differently, which is possible as the model is infinite).

Then run the construction from the completeness proof to find a Henkin extension $\hat{\Gamma}$ of Γ^*. The construction is repeated here:

- Base case. $\mathcal{L}_0 = \mathcal{L}^*$ and $\Gamma_0 = \Gamma^*$.
- Recursive step. Assume we have defined $\mathcal{L}_n$ and Γ_n.

  For each $A \in \mathcal{L}_n$ let c_A be a symbol not used in $\mathcal{L}_n$ and define $\mathcal{L}_{n+1} = \langle \mathcal{C}_{n+1}, \mathcal{F}_{n+1}, \mathcal{R}_{n+1} \rangle$ as follows:

  $$\mathcal{C}_{n+1} = \mathcal{C}_n \cup \{c_A \mid A \in \texttt{wff}(\mathcal{L}_n)\}, \mathcal{F}_{n+1} = \mathcal{F}_n, \text{ and } \mathcal{R}_{n+1} = \mathcal{R}_n.$$

  Let $\Gamma_{n+1} \subseteq \texttt{wff}(\mathcal{L}_{n+1})$ be a maximally consistent superset of the following set:

  $$\Gamma_n \cup \{\neg A[c_A/x] \mid A \in \texttt{wff}(\mathcal{L}_n) \text{ and } x \in \mathcal{V} \text{ and } \neg\forall x A \in \Gamma_n\}.$$

Let $\hat{\Gamma} = \bigcup_{n\in\mathbb{N}} \Gamma_n$, $\hat{\mathcal{L}} = \langle \bigcup_{n\in\mathbb{N}} \mathcal{C}_n, \mathcal{F}, \mathcal{R} \rangle$ and let M, I, f be the term model. As in the proof of completeness, $M, I, f \models B$ for all $B \in \Gamma^*$. Since this means that the c_i all receive different interpretations, M must be infinite. We can now show that M is countable.

Recall that unions of countably many countable sets are countable (Theorem 2.28) and that this means that over a countable language we only have countably many formulae (can be shown analogously to Theorem

2.29). This means in every stage of the Henkin recursion, we only add countably many new constant symbols from which, again, only countably many formulae can be formed. Thus $\bigcup_{n\in\mathbb{N}}\mathcal{C}_n$ is countable as well. Thus, because $\mathcal{F}$ and $\mathcal{R}$ are countable, $\hat{\mathcal{L}}$ is countable and so only countably many terms can be formed in that language.

But we know that $M = \{[t] \mid t \in \mathtt{Tm}(\hat{\mathcal{L}})$, so M has as most as many members as there are terms. Thus M is countable. As M is also infinite, M is countably infinite. This was to show. □

This result has a surprising consequence. Let $\mathcal{L}$ be the language of set theory and **ZFC** be the standard set of set theoretic axioms. The Countable Model Theorem entails that there is a countable model of **ZFC**. *But* from the **ZFC** axioms one can prove Cantor's theorem. So there is a countable set (a countable model of **ZFC**) that 'thinks' it has uncountably large members (because Cantor's theorem is true within the model). This is known as **Skolem's paradox**.

Skolem's paradox was very disturbing to logicians in the early 20th century, before the expressive limitations of the first-order language were well understood. Today, we can give a fairly glib explanation of why Skolem's paradox should not disturb us. The claim 'there is an uncountable set' has an essential *negative existential* component: it says that there is a set s and *there isn't* a one-to-one function from s to the natural numbers. This can be true in a very small model if it simply does not contain enough material to build the one to one function. The small model 'thinks' a set that is actually countable is uncountable because it is too impoverished to contain the one-to-one function.

The Countable Model Theorem is a special case of a result covering larger languages.

Theorem 6.6 (Löwenheim-Skolem Theorem). *Let $\mathcal{L}$ be any language and let Γ be a set of sentences that has an infinite model. Then for every infinite set s with $|s| \geq |\mathcal{L}|$, Γ has a model of size $|s|$.*

(In some textbooks, already the Countable Model Theorem is called the Löwenheim-Skolem Theorem.)

Thus, we cannot control at all the sizes of infinite models. And in larger languages, only the language itself is a lower bound.

We will use another fact from Set Theory.

Fact 6.7. *Let s be a set of infinite size and let $\{s_i \mid i \in \mathbb{N}\}$ be countably many sets such that $|s| = |s_i|$ for all $i \in \mathbb{N}$. Then $\left|\bigcup_{i\in\mathbb{N}} s_i\right| = |s|$.*

This allows us to prove the Löwenheim-Skolem Theorem similarly to how we proved the Countable Model Theorem. We will again only look at an extended sketch of the argument.

Proof of Löwenheim-Skolem. Let $\mathcal{L}$ be any language and let Γ be a set of sentences that has an infinite model M, I, f. Let s be any infinite set with $|s| \geq \mathcal{L}$.

For every $x \in s$ let c_x be an unused constant symbol. Let $\mathcal{L}^*$ be the extension of $\mathcal{L}$ with these new symbols.

Consider the following set Γ^*.

$$\Gamma^* = \Gamma \cup \{\neg c_x \equiv c_y \mid x \in s, y \in s \text{ and } x \neq y\}.$$

Show that Γ^* is satisfiable. By the Compactness Theorem, it suffices to show that every finite $\Gamma' \subseteq \Gamma$ is satisfiable. So let $\Gamma' \subseteq \Gamma^*$ be finite and otherwise arbitrary.

As Γ' is finite, it contains only finitely many formulae of the form $\neg c_x \equiv c_y$ where $x \in s$ and $y \in s$. We can extend M, I, f to a model of Γ' by assigning to each c_x some element of M, making sure that the (finitely many) such c_x that occur anywhere in Γ' are assigned distinct members of M (which is possible, as M is infinite by assumption).

This shows that all finite subsets of Γ^* are satisfiable, so Γ^* is satisfiable. In particular, Γ^* is consistent (that satisfiability entails consistency follows from Soundness: if $\Gamma^* \vdash \bot$, the n$\Gamma^* \models \bot$, so Γ^* is not satisfiable.). Then let M^*, I^*, f^* be the term model of Γ^*.

As we did in the proof of the Countable Model Theorem, we can compute the size of M^*. By the previous Fact, there are $|s|$-many wffs over $\mathcal{L}$, so in every step of Henkin's construction, we add $|s|$-many new constants which means (by an induction argument) that after completing the construction, we have a term model over a language of size $|s|$. Thus the term model has at most $|s|$ members. As the added formulae of the form $\neg c_x \equiv c_y$ ensure that we have at least $|s|$-many members, $|s| = |M^*|$. This was to show. □

Sometimes, the Löwenheim-Skolem Theorem is separated into a 'downward' and 'upward' part and is phrased in terms of models instead of sets of sentences.

Theorem 6.8 (Upward Löwenheim-Skolem Theorem). *Let $\mathcal{L}$ be any language M, I, f be an infinite model in L. Then for every infinite set s with $|s| \geq \mathcal{L}$ and $|s| \geq |M|$, there is a model M', I', f' with $|M'| \geq |s|$ and $\mathrm{Th}(M, I, f) = \mathrm{Th}(M', I', f')$.*

Theorem 6.9 (Downward Löwenheim-Skolem Theorem). *Let $\mathcal{L}$ be any language M, I, f be an infinite model in L. Then for every infinite set s with $|s| \geq \mathcal{L}$ and $|s| \leq |M|$, there is a model M', I', f' with $|M'| = |s|$ and $\mathrm{Th}(M, I, f) = \mathrm{Th}(M', I', f')$.*

The proof of both follows from using the proof of Theorem 6.6 for the set $\Gamma = \mathrm{Th}(M, I, f)$.

7 Incompleteness

The goal of this section will be to show the following.

Theorem 7.1 (Gödel's First Incompleteness Theorem). *If $\mathbf{PA}$ is consistent, then there is a sentence G in the language of arithmetic such that neither $\mathbf{PA} \vdash G$ nor $\mathbf{PA} \vdash \neg G$.*

The rough, intuitive meaning of the sentence G will be 'G is not provable in Peano Arithmetic'. But to obtain such a G, we first need to express *provability in PA* in the language of arithmetic. The first step is to formalise the language of arithmetic *from within the language of arithmetic* and then use a remarkable trick to find the sentence G.

7.1 The Arithmetisation of Syntax

As noted multiple times throughout this course, it fundamentally does not matter what the symbols of our language are. The language of arithmetic $\mathcal{L}^A$ contains a constant symbol 0 and three function symbols s, $+$. We then construct formulae from the additional symbols $\equiv$,), (, $\neg$, $\wedge$, $\vee$, $\rightarrow$, $\forall$, $\exists$ and countably infinitely many variable symbols. Instead of these symbols, we might as well use *numbers* to form formulae.

- Instead of the symbol 0, use the number 2.
- Instead of the symbol s, use the number 3.
- Instead of the symbol $+$, use the number 4.
- Instead of the symbol $\cdot$, use the number 5.
- Instead of the symbol $\equiv$, use the number 6.
- Instead of the symbol), use the number 7.
- Instead of the symbol (, use the number 8.
- Instead of the symbol $\neg$, use the number 9.
- Instead of the symbol $\wedge$, use the number 10.
- Instead of the symbol $\vee$, use the number 11.
- Instead of the symbol $\rightarrow$, use the number 12.
- Instead of the symbol $\forall$, use the number 13.
- Instead of the symbol $\exists$, use the number 14.
- Instead of variable symbols $x_0, x_1, \ldots$ use numbers larger than 14.

If *this* is our logical language, then the first Robinson axiom $(\forall x_0(\neg s x_0 \equiv 0))$ would be written as the sequence $\langle 13, 15, 8, 9, 3, 15, 6, 2, 7\rangle$. This is a lot harder to read *for us*, but since the symbols carry no inherent meaning, it would not change anything about what we have done so far.

We now have an alphabet of symbols that we can talk about in the language of arithmetic, but to speak of *formulae* we need to be able to speak of *sequences* of symbols. Unfortunately, the language of arithmetic does not contain a mechanism to form ordered sets. But as discovered by Gödel, we can speak of certain (very large) numbers as *containing all the information* that we would normally encode in an ordered set. Despite being a substantial insight, this is not magic. The following has been known since antiquity.

Fact 7.2 (Fundamental Theorem of Arithmetic)**.** *For every number n, there is a unique i and a unique sequence $\langle e_0, \ldots, e_{i-1}\rangle$ such that $n = p_0^{e_0} \cdot p_1^{e_1} \cdot \ldots \cdot p_{i-1}^{e_{i-1}}$ where for each k, p_k is the kth prime number.*

That is, we can write every number n as as a product of prime numbers, where it is uniquely determined how often any prime number occurs in the product. (Note that if for some k, the kth prime number p_k does not occur at all in the product, then $e_k = 0$.)

Put differently, writing numbers as the sequence of exponents of their factorisation into primes is *uniquely readable*. That is, the information of the sequence $\langle e_0, ..., e_{i-1}\rangle$ is 'encoded' (if you will) in the number n that can be written as $n = p_0^{e_0} \cdot ... \cdot p_{i-1}^{e_{i-1}}$. From the sequence, we can get n by simple multiplication and from n we can get the sequence by factorisation.

Thus, for all intents and purposes, n and $\langle e_0, ..., e_{i-1}\rangle$ are interchangeable. We could actually decide to 'think' of numbers as being sequences of numbers. That is, we *could* define a model of arithmetic in which all members are sequences of natural numbers and define the standard interpretation of s, $+$ and $\cdot$ for these sequences. In this model, we would have the sequence $\langle 13, 15, 8, 9, 3, 15, 6, 2, 7\rangle$ as a member, and we can treat ('think of') this sequence as the formula $\forall x_0(\neg sx_0 \equiv 0)$.

But this is not quite sufficient. We want to be able to treat numbers as sequences from within *Peano Arithmetic*. That is, we want to be able to *prove* in $\mathbf{PA}$ things like 'm is the ith member of the sequence n'.

Encouragingly, it is indeed the case that we can express things like 'm is the ith member of the sequence n' by only using arithmetical operations. To wit, we may say that the number m is the ith member of the sequence n iff for p_i being the ith prime number, it is the case that p_i^m is a divisor of n and p_i^{m+1} is not a divisor of n.

The following definitions make clear what we are demanding from Peano Arithmetic.

Definition 61. Given a natural number n, it's **canonical name** is the term $s^n 0$. For brevity, we write $\mathbf{n}$ as an abbreviation of the canonical name of n.

That is, e.g., $\mathbf{2}$ is an abbreviation of $ss0$, $\mathbf{5}$ is an abbreviation of $sssss0$ and so on.

Definition 62 (Representation)**.** Let n be a number and $P : \mathbb{N}^n \to \{0, 1\}$ be an n-ary property of the numbers. Say that P is **representable in PA** iff$_{\text{Def}}$ there is a formula $A^P(x_0, ..., x_{n-1})$ in the language of arithmetic such that for all numbers $m_0, ..., m_{n-1}$ it is the case that:

$$\mathbf{PA} \vdash A^P[\mathbf{m}_0, ..., \mathbf{m}_{n-1}/x_0, ..., x_{n-1}] \text{ iff } P(m_0, ..., m_{n-1}) = 1.$$

Note that this only makes sense if $\mathbf{PA}$ is indeed consistent (which we are tacitly assuming here). Because if it is inconsistent, then $\mathbf{PA} \vdash A^P[\mathbf{m}_0, ..., \mathbf{m}_{n-1}/x_0, ..., x_{n-1}]$ for *all* numbers $m_0, ..., m_{n-1}$ and not just for the ones with property P. (If we are worried about $\mathbf{PA}$ possibly being inconsistent, we can use the same definition and just use the right-to-left part of the 'iff'.)

That is, a property is representable in Peano Arithmetic when for all numbers it follows from $\mathbf{PA}$ that they (their canonical names) have the property exactly when they actually do have that property.

Remark: The definition of Representation does not ensure that weird things can't happen for non-standard numbers. For example, it could be the case that all numbers have some unary property P that is representable, i.e. for all numbers n, $\mathbf{PA} \vdash A^P[\mathbf{n}/x_0]$. It does *not* follow that $\mathbf{PA} \vdash \forall x_0 A^P$. If $\mathbf{PA}$ does not prove the universally quantified statement, then there is a non-standard model containing a non-standard number a that does not have the property A^P (in that model). We would then say that the non-standard number does not have the property P, but this is *loose talk* as we only really understand what P means for the standard numbers. Such non-standard numbers do not contradict the representability of P.

Theorem 7.3 (Gödel)**.** *The relation 'm is the ith member of n' is representable in Peano Arithmetic.*

For the proof, we will assume that we already have the following.

- There is a formula *Pr*(n, m) that represents 'm is the nth prime number'.
- There is a formula *Ex*(n, m, i) that represents 'i is n^m'.

These are *very* hard to find and we shall make no attempt to do so here. The problem is that it is very easy to find a formula $\mathit{Pr}_2(m)$ saying that m is the 2nd prime: with a predicate *prime* and a defined relation $<$ in place, this is just:

$$\mathit{prime}(m) \land \exists k(k < m \land \mathit{prime}(k) \land \forall i((i < m \land \mathit{prime}(i)) \to k \equiv i)$$

Similarly, we can express that m is the 3rd prime:

$$\mathit{prime}(m) \land \exists k \exists j(k < m \land j < k \land \mathit{prime}(k) \land \mathit{prime}(j) \land \forall i((i < m \land \mathit{prime}(i)) \to (k \equiv i \lor j \equiv i))$$

But it is very *hard* to generalise this to find *Pr*(m, n), as the number of quantifiers we need to say 'nth prime' appears to depend on n. Put differently, expressing *in general* that 'there are $n - 1$ primes below m' is talk about a *set* of numbers, but our quantifiers only talk about first-order numbers. Gödel solved this problem by using a more sophisticated method of coding sequences than we are using, using more deep mathematical results than the Fundamental Theorem. We will not replicate this method here.

Proof of Theorem 7.3. Using *Pr* and *Ex*, we can state a formula *seq*(n, i, m) that represents 'm is the ith member of n'. To wit:

$$\mathit{seq}(n, i, m) = \forall p \forall q \forall r(\mathit{Pr}(i, p) \to ((\mathit{Ex}(p, m, q) \to \exists k(k \cdot q \equiv n)) \land (\mathit{Ex}(p, sm, r) \to \neg \exists k(k \cdot r \equiv n)))).$$

That is, *seq*$[\mathbf{n}, \mathbf{i}, \mathbf{m}/n, i, m]$ is the case in the intended interpretation of arithmetic if the ith prime number (i.e. p), taken to the mth power (i.e. q) is a divisor of n (i.e. part of its prime factorisation) and the ith prime number (i.e. p) taken to the $m + 1$st power (i.e. r) is not a divisor of n. But this is indeed just the case if the ith member of n is m.

Given what *Pr* and *Ex* represent and that $\mathbf{PA}$ proves all simple arithmetical facts (such as when $k \cdot r = n$), it is easy to see that $\mathbf{PA}$ proves *seq*$[\mathbf{n}, \mathbf{i}, \mathbf{m}/n, i, m]$ exactly when it is true. □

Thus, by using the formula *seq* we can extract the 'information' about this large number being a sequence *from within Peano Arithmetic.* With this, we can start constructing all we need to do logic from within arithmetic. (We will not spell out the language-internal definitions in detail, instead being convinced by the following remarks.)

- There is a formula *len*(n, m) representing 'm is the length of n (when understood as a sequence)'. To define this, we define what it means to be the largest prime number that divides n and let m be the index of this prime number (defined by *Pr*).
- There is a formula *cat*(n, m, k) representing 'k is the concatenation of n and m (when all three are understood as sequences)'. We can find this formula by letting i be the length of n, 'decomposing' m into its prime factors $p_0^{e_0} \cdot \ldots \cdot p_{n-1}^{e_{n-1}}$, computing $p_{i+0}^{e_0} \cdot \ldots \cdot p_{i+n-1}^{e_{n-1}}$ and multiplying this with n.
- There is a formula *term*(k) representing 'k is a term in the language of arithmetic' (with the symbols of the language being as above). We can obtain this formula as follows: k is a term iff there is a sequence n such that the last member of n is k and every member of n is either 2 (i.e. the constant symbol for 0) or a number larger 14 (i.e. a variable symbol) or formed by concatenating $\langle 3 \rangle$ (i.e. s) with previous member of n or concatenating $\langle 4 \rangle$ or $\langle 5 \rangle$ (i.e. $+$ or $\cdot$) with two previous members of n.

The method for *term*(k) shows how to use our method of 'thinking' of formulae as sequences and of sequences as numbers can be used to give recursive definitions *in the language of arithmetic*. We can say that a number k is obtained by a recursion if there is a sequence n that ends in k and every previous entry in the sequence n is either an atom or a recursive step on two previous members.

- Using the representation of concatenation, we can represent the formation of formulas. For example, there is a formula *cond*(i, j, k) representing that the sequence coded by k is obtained by forming $(s_1 \to s_2)$, where s_1 is the sequence coded by i and s_2 is the sequence coded by j.
- Then, there is a formula *wff*(k) representing 'k is a formula in the language of arithmetic'. We get this by recursion like the terms.
- There is a formula *Ax*(k) representing 'k is an axiom of Peano Arithmetic'. The only problem is the induction scheme, but having defined *wff*, we can easily define what it means to be an instance of the induction scheme.

Now, given the association of numbers with symbols at the beginning of this section, we can define the following (in the metalanguage).

Definition 63 (Gödel Codes)**.** Let A be a formula in the language of arithmetic. Define $\ulcorner A \urcorner$ to be the natural number whose prime exponents are A (in the language where the symbols are numbers). $\ulcorner A \urcorner$ is the **Gödel code** of A.

For example, we saw that we can represent $\forall x_0(\neg s x_0 \equiv 0)$ by the sequence $\langle 13, 15, 8, 9, 3, 15, 6, 2, 7 \rangle$. Taking these as prime exponents, we get $\ulcorner \forall x_0(\neg s x_0 \equiv 0) \urcorner = 2^{13} \cdot 3^{15} \cdot 5^{8} \cdot 7^{9} \cdot 11^{3} \cdot 13^{15} \cdot 17^{6} \cdot 19^{2} \cdot 23^{7}$. This is a very large number, but it contains the 'same information' as the sequence $\langle 13, 15, 8, 9, 3, 15, 6, 2, 7 \rangle$.

Given a Gödel code $\ulcorner A \urcorner$, we will now write $\mathbf{n}^A$ for the canonical name for the number $\ulcorner A \urcorner$ that codes the formula A. That is, if $A = \forall x_0(\neg s x_0 \equiv 0)$, then $\mathbf{n}^A$ is an abbreviation of the term containing $2^{13} \cdot 3^{15} \cdot 5^{8} \cdot 7^{9} \cdot 11^{3} \cdot 13^{15} \cdot 17^{6} \cdot 19^{2} \cdot 23^{7}$ many s, followed by 0.

7.2 The Fixed Point Lemma

The crucial result for showing Incompleteness is the following technical result.

Theorem 7.4 (Fixed Point Lemma)**.** *For every formula $A(x)$ in the language of arithmetic (with exactly one free variable) there is a closed formula D such that:*

$$\mathbf{PA} \vdash (D \to A[\mathbf{n}^D/x]) \wedge (A[\mathbf{n}^D/x] \to D),$$

which we will abbreviate as $\mathbf{PA} \vdash (D \leftrightarrow A[\mathbf{n}^D/x])$.

The Fixed Point Lemma says that there is a formula D that is equivalent to $\ulcorner D \urcorner$ having the property A. *Very roughly*, one sometimes says that the meaning of D is 'I have property A' (which of course is inaccurate in many ways, as D merely says that some number $\ulcorner D \urcorner$ that we have *associated in an entirely arbitrary way* with D has the property A). It's called the *Fixed Point Lemma* because a fixed point of an operation is something that remains unchanged under the operation. In a sense, D is a fixed point of the 'operation' of *applying* A: the truth value of D remains unchanged by applying A to 'it' (properly: to the number we have arbitrarily designated as its code).

This result is also known as the **diagonal lemma** because the proof involves an operation whereby one maps a formula $B(z)$ to $B[\mathbf{n}^B/z]$. Plugging something into itself is typically called *diagonalising*.

Proof. We begin by defining what it means to apply a formula (given by its code) to another formula (given by its code). Define a function $f : \mathbb{N}^2 \to \mathbb{N}$ by:

$$f(n,m) = \begin{cases} \ulcorner B[\mathbf{m}/z] \urcorner, \text{ if } n = \ulcorner B(z) \urcorner \text{ for some formula } B(z) \text{ in a single free variable} \\ 0, \text{ if } n \text{ is not code of a formula in a single free variable.} \end{cases}$$

If $Z(x)$ is a formula in a single free variable then we call the output of $f(\ulcorner Z \urcorner, \ulcorner Z \urcorner)$ **the diagonal of** Z (i.e. Z "plugged into itself").

With the work we have done on coding, we can write a formula $F(x_2, x_1, x_0)$ that represents the property 'x_0 is the output of $f(x_2, x_1)$'. That is, $\mathbf{PA} \vdash F[\mathbf{m}, \mathbf{n}, \mathbf{k}/x_2, x_1, x_0]$ iff $f(m,n) = k$ (in the meta-language).

Now let A be an arbitrary formula in the language of arithmetic with x_0 being its sole free variable. Consider the following formula that has a single free variable z:

$$B = \forall x_0 (F[z, z/x_2, x_1] \to A)$$

This is a property of (codes of) sentences: "the code for the sentence *the code of Z has property* Z has property A'. (Or: 'the diagonal of Z has property A'.)

Using the representation of f we can plug B into itself.

$$D = \forall x_0 (F[\mathbf{n}^B, \mathbf{n}^B/x_2, x_1] \to A)$$

This is the formula that says 'the code for the sentence *the code of B has property B* has property A'. (Or: 'the diagonal of B has property A.)

Note that $D = B[\mathbf{n}^B/z]$, which means that $f(\ulcorner B \urcorner, \ulcorner B \urcorner) = \ulcorner D \urcorner$. That is the code of *the code of B has property B* **is just the code for** D. Thus D indeed says that its own code has property A. (Or: D says that the diagonal of B has property A and *its own* code $\ulcorner D \urcorner$ is the diagonal of B.)

It remains to show this formally. Because F represents f it is the case that:

$$\mathbf{PA} \vdash F[\mathbf{n}^B, \mathbf{n}^B, \mathbf{n}^D/x_2, x_1, x_0].$$

Now we can show that Peano Arithmetic proves that D is equivalent to $A[\mathbf{n}^D/x_0]$.

1. Show that $\mathbf{PA} \vdash D \to A[\mathbf{n}^D/x_0]$.

   By definition of D and (∀E.), $\mathbf{PA} \cup \{D\} \vdash F[\mathbf{n}^B, \mathbf{n}^B/x_2, x_1][\mathbf{n}^D/x_0]) \to A[\mathbf{n}^D/x_0]$. Write this more compactly as:

   $$\mathbf{PA} \cup \{D\} \vdash F[\mathbf{n}^B, \mathbf{n}^B, \mathbf{n}^D/x_2, x_1, x_0] \to A[\mathbf{n}^D/x_0]$$

   Note that we have the antecedent of this. So by (→E.).

   $$\mathbf{PA} \cup \{D\} \vdash A[\mathbf{n}^D/x_0]$$

   Thus by (→I.):

   $$\mathbf{PA} \vdash D \to A[\mathbf{n}^D/x_0]$$

2. Now show that $\mathbf{PA} \vdash A[\mathbf{n}^D/x_0] \to D$.

   Note that because f is a function we have the following.

   $$\mathbf{PA} \vdash \forall i \forall j ((F[\mathbf{n}^B, \mathbf{n}^B, i/x_2, x_1, x_0] \land F[\mathbf{n}^B, \mathbf{n}^B, j/x_2, x_1, x_0]) \to i \equiv j)$$

   This means by (∀E.) and because $\mathbf{PA} \vdash F[\mathbf{n}^B, \mathbf{n}^B, \mathbf{n}^D/x_2, x_1, x_0]$ that:

   $$\mathbf{PA} \vdash \forall i (F[\mathbf{n}^B, \mathbf{n}^B, i/x_2, x_1, x_0] \to i \equiv \mathbf{n}^D).$$

   By a simple proof (involving the substitution of identicals) this means:

   $$\mathbf{PA} \vdash A[\mathbf{n}^D/x_0] \to \forall i (F[\mathbf{n}^B, \mathbf{n}^B, i/x_2, x_1, x_0] \to A[i/x_0]).$$

   By re-binding the quantified variable i to x_0:

   $$\mathbf{PA} \vdash A[\mathbf{n}^D/x_0] \to \forall x_0 (F[\mathbf{n}^B, \mathbf{n}^B/x_2, x_1] \to A).$$

But the right-hand side of this just is D. Thus:

$$\mathbf{PA} \vdash A[\mathbf{n}^D/x_0] \to D.$$

This concludes the proof. □

7.3 The Incompleteness Theorem

The Fixed Point Lemma is very powerful. It allows us to generate for any property a sentence that (roughly, informally) says of itself that it does not have the property. Such sentences are sometimes called *diagonals* or *diagonalising* the property. Recall that to show the incompleteness theorem, we wanted a sentence G that (roughly) says of itself that it is not provable. We get this from the fixed point lemma if we can represent provability in $\mathbf{PA}$.

Theorem 7.5. *The property* x being a code for a formula provable from PA *is representable in* $\mathbf{PA}$ *by a formula* $\mathit{Pf}_{\mathbf{PA}}(x)$.

Proof. Use the Hilbert calculus for characterising $\vdash$. Then proofs are sequences. So we can say that $\mathit{Pf}_{\mathbf{PA}}(x)$ is the formula that says there is a formula y coding a sequence such that:

- x codes a wff and all members of y code wffs.
- If *len*(y, i), then *seq*(y, i, x) (i.e. x is the last element of y).
- For each $j \leq i$ and all k: if *seq*(y, j, k), then either
    - k is (a code for) a Peano Axiom; or
    - k is (a code for) a substitution instance of a logical axiom; or
    - there are $l < j$ and $o < j$ such that if n is the lth entry in y and n is the oth entry in y, then *cond*(n, k, m) (i.e. k is obtained by *modus ponens* from n and m).

We already saw that we can represent 'being a Peano Axiom' and 'being a conditional'. It is not difficult (but tedious) to represent substitution and so we can represent 'substitution instances of logical axioms'.

Another, perhaps easier way to obtain this result is to realise that we can represent Turing computability in $\mathbf{PA}$ and then find a Turing machine that outputs all proofs. Then $\mathit{Pf}_{\mathbf{PA}}(x)$ can be defined as there being a number n such that this TM outputs a proof with conclusion x in step n of its computation. (This would be establishing/exploiting the fact that the set of proofs from $\mathbf{PA}$ is *recursively enumerable.*) □

Be careful with the representation of provability. It is not the case, for example, that $\mathbf{PA} \nvdash A$ entails that $\mathbf{PA} \vdash \neg \mathit{Pf}[\mathbf{n}^A/x]$. This is because by definition of representation, $\mathbf{PA} \nvdash A$ only entails that it is *not* the case that $\mathbf{PA} \vdash \mathit{Pf}(\mathbf{n}^A/x]$. Something *not* being provable from Peano Arithmetic does not entail that Peano Arithmetic proves its negation. This is because $\mathbf{PA}$ is incomplete, as we can now demonstrate.

Theorem 7.6 (Gödel's First Incompleteness Theorem)**.** *If* $\mathbf{PA}$ *is consistent, then there is a sentence* G *in the language of arithmetic such that neither* $\mathbf{PA} \vdash G$ *nor* $\mathbf{PA} \vdash \neg G$.

Proof. Using the Fixed Point Lemma we can find a sentence G such that $\mathbf{PA} \vdash G \leftrightarrow \neg \mathit{Pf}_{\mathbf{PA}}[\mathbf{n}^G/x]$. Assume that $\mathbf{PA}$ is consistent and show that neither $\mathbf{PA} \vdash G$ nor $\mathbf{PA} \vdash \neg G$.

- Assume for *reductio* that $\mathbf{PA} \vdash G$. Then there is a proof of G from Peano Arithmetic and thus there is a natural number n that codes that proof. Because $\mathit{Pf}_{\mathbf{PA}}$ represents provability in $\mathbf{PA}$, it follows that $\mathbf{PA} \vdash \mathit{Pf}_{\mathbf{PA}}[\mathbf{n}^G/x]$.

  But by definition of G, $\mathbf{PA} \vdash G$. So by (→E.), $\mathbf{PA} \vdash \neg \mathit{Pf}_{\mathbf{PA}}[\mathbf{n}^G/x]$. Thus by (¬I.), $\mathbf{PA} \vdash \bot$. Contradiction to the assumption that $\mathbf{PA}$ is consistent. So by *reductio*, $\mathbf{PA} \nvdash G$.

- Assume for *reductio* that $\mathbf{PA} \vdash \neg G$. This means by (Contraposition) and (→E.) that $\mathbf{PA} \vdash \mathit{Pf}_{\mathbf{PA}}(\mathbf{n}^G)$. Let $\mathbb{N}, I$ be the standard interpretation of arithmetic and f be any assignment. As Peano Arithmetic is true in the standard interpretation, $\mathbb{N}, I, f \models \mathit{Pf}_{\mathbf{PA}}(\mathbf{n}^G)$. This means that there is a natural number (i.e. a member of $\mathbb{N}$) that codes a proof of G. Using our work on coding, we can 'decode' to find the actual proof. But then $\mathbf{PA} \vdash G$, so by (¬I.), $\mathbf{PA} \vdash \bot$. Contradiction to $\mathbf{PA}$ being consistent. So by *reductio*, $\mathbf{PA} \nvdash \neg G$.

This concludes the proof. □

Remark: The assumption that $\mathbf{PA}$ is true in the standard interpretation is slightly stronger than $\mathbf{PA}$ being consistent (Gödel called this assumption *ω-consistency*). Using a technique called *Rosser's Trick* one can do the proof without relying on the standard interpretation.

Note that since $\mathbf{PA} \nvdash G$ no actual natural number can be the code of a proof of G. So in the standard model of arithmetic, G is true. So the fact that G is not provable must mean that there is some other model of $\mathbf{PA}$ in which there is a code for a proof of $\ulcorner G \urcorner$. This can only be a non-standard number. In a sense, this non-standard number codes a 'non-standard proof' of G.

Perhaps more famous than the First Incompleteness Theorem is the Second one, stating that $\mathbf{PA}$ cannot prove its own consistency.

Theorem 7.7 (Gödel's Second Incompleteness Theorem)**.** *If* $\mathbf{PA}$ *is consistent, then it is not the case that* $\mathbf{PA} \vdash \neg \mathit{Pf}_{\mathbf{PA}}[\mathbf{n}^{\bot}/x]$.

The sentence $\neg \mathit{Pf}_{\mathbf{PA}}[\mathbf{n}^{\bot}/x]$ represents $\mathbf{PA} \nvdash \bot$, i.e. that $\mathbf{PA}$ is consistent.

Proof. Assume $\mathbf{PA}$ is consistent. We can formalise the proof of the First Incompleteness Theorem *within* the coding language. This is very tedious and a bit of work, but the process is not particularly illuminating. We take codes of our formulae for coding like *seq* and can then talk about codes of codes. This allows us to phrase and prove in $\mathbf{PA}$ the object-language version of the Fixed Point Lemma (roughly: for all codes of open formulae a there is a code for a formula d such that $\mathit{Pf}(b)$, where b codes the biconditional of a and plugging d into a).

Having done this, we obtain the following as the object language version of the First Incompleteness Theorem.

$$\mathbf{PA} \vdash \neg \mathit{Pf}_{\mathbf{PA}}[\mathbf{n}^{\bot}/x] \to \neg \mathit{Pf}_{\mathbf{PA}}[\mathbf{n}^G/x].$$

Now assume for *reductio* that $\mathbf{PA} \vdash \neg \mathit{Pf}_{\mathbf{PA}}[\mathbf{n}^{\perp}/x]$, i.e. that Peano Arithmetic proves its own consistency. Then $\mathbf{PA} \vdash \neg \mathit{Pf}_{\mathbf{PA}}[\mathbf{n}^{G}/x]$ by (→E.). By definition of G, this means that $\mathbf{PA} \vdash G$. But by the First Incompleteness Theorem, this is not the case. Contradiction. Hence by *reductio*, it is not the case that $\mathbf{PA} \vdash \neg \mathit{Pf}_{\mathbf{PA}}[\mathbf{n}^{G}/x]$. □

Evidently, nothing here hinged on us talking about Peano Arithmetic *specifically*. We can conclude that every theory that can represent its own provability relation is incomplete and cannot prove itself consistent. This notably includes $\mathbf{Q}$ (as shown by RM Robinson) and $\mathbf{ZFC}$.

Note that if a superset of $\mathbf{PA}$ is definable by a purely syntactic, recursive definition (as $\mathbf{PA}$ itself), it can represent its own provability relation (as we can combine the definition of the axioms with the definition of proof, as above). This suffices to run all the same arguments again. So trying to extend $\mathbf{PA}$ to 'make it complete' is hopeless.

In particular, it follows that there is no recursive set of axioms from which we can derive all members of $\mathbf{TA}$, as any such axiomatisation is incomplete, but True Arithmetic is complete. Thus:

Theorem 7.8. *True Arithmetic is not recursively axiomatisable.*

7.4 The Undefinability of Truth

The previous results hinged on the representability of the provability relation. The following result does not require this.

Theorem 7.9 (Tarski's Undefinability Theorem)**.** *The property* being a member of $\mathbf{TA}$ *is not representable in* $\mathbf{TA}$.

Proof. Assume for *reductio* there is a formula $T(x)$ such that $\mathbf{TA} \vdash T(\mathbf{n})$ iff there is a formula A such that $n = \ulcorner A \urcorner$ and $A \in \mathbf{TA}$. Then, by the Fixed Point Lemma, there is a sentence L (a 'liar') such that:

$$\mathbf{TA} \vdash L \leftrightarrow \neg T[\mathbf{n}^{L}/x].$$

Because $\mathbf{TA}$ is complete, we are in one of two cases.

- Case 1. $L \in \mathbf{TA}$. Then by definition of T, $\mathbf{TA} \vdash T[\mathbf{n}^{L}/x]$. But by definition of L and by (→E.) $\mathbf{TA} \vdash \neg T[\mathbf{n}^{L}/x]$. But this means that $\mathbf{TA}$ is inconsistent.
- Case 2. $\neg L \in \mathbf{TA}$. Then by definition of L and (→E.) $\mathbf{TA} \vdash T[\mathbf{n}^{L}/x]$. By definition of T, then $L \in \mathbf{TA}$. But this means that $\mathbf{TA}$ is inconsistent.

Thus in both cases we conclude that $\mathbf{TA}$ is inconsistent. But it is not, because it has a model. Contradiction. Thus there is no such T. □

This uses less resources than Gödel's theorems: any complete theory that has the resources required for the Fixed Point Lemma (essentially just the coding of formulae in sequences and the representation of some functions on codes) is subject to this result and hence cannot represent its own membership. (The stronger requirement for Gödel's theorems is the representability of proofs.) But for theories that do not have these resources, it doesn't even make sense to talk about representing its own membership (how would that look like, if we can't represent formulae?).

Tarski concluded that one cannot express the truth predicate of an object language from within the object language, but must instead ascend to a meta-language. This is the 'ultimate' result in our ongoing theme about the superior expressibility of the meta-language.

Another immediate consequence from Undefinability is this:

Theorem 7.10. *There is no Turing Machine that, when run, sequentially outputs all and only members of* $\mathbf{TA}$.

Proof. As said, Turing Machines can be represented in arithmetic. If there were such a TM as in the theorem, it could be used to represent membership in $\mathbf{TA}$, which is impossible *per* the previous result. □

This is roughly why Incompleteness is sometimes (somewhat misleadingly and *much* too vaguely) paraphrased as 'there is always something that we cannot know'. This last result means that there is no 'process' by which we can generate all arithmetical truths.