A Lean Systematization of Mathematical Logic

From the Open Logic Project

Every concept traceable to primitives.
No redundancy. Full dependency graph.

Based on *Open Logic: A Complete Text*
by Richard Zach and the Open Logic Project Contributors

Preface

What This Book Is

This book presents the core of mathematical logic—from naive set theory through Gödel's incompleteness theorems—as a single, self-contained development in which every concept is defined exactly once, every theorem is traceable to its primitives, and the dependency structure is explicit throughout.

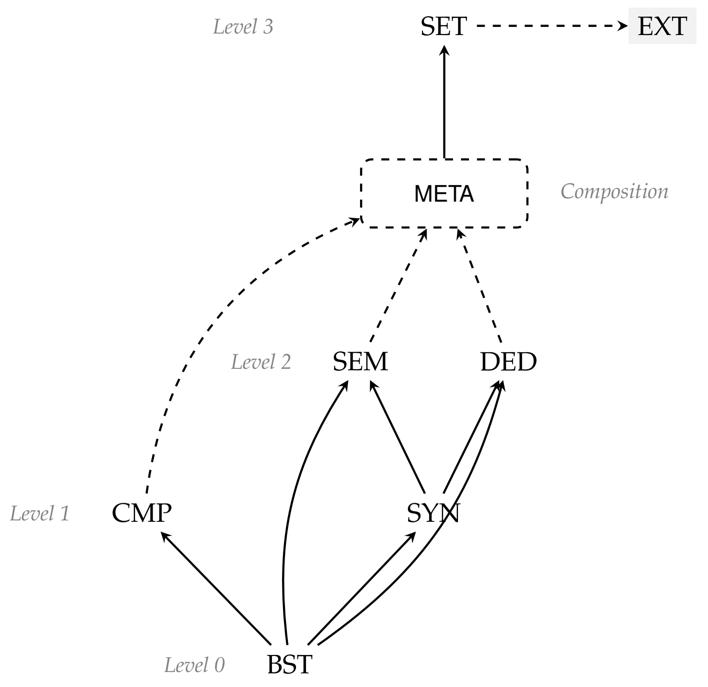
The material is drawn from the Open Logic Project's *Open Logic: A Complete Text*, a collaboratively authored, open-source textbook covering propositional and first-order logic, model theory, computability, incompleteness, and set theory. That text is modular by design, offering instructors flexibility to assemble custom courses. The present volume takes the opposite approach: it *compresses* the source material into a single linear narrative organized by a taxonomy of mathematical logic, eliminating redundancy and making every dependency visible.

How It Is Organized

The taxonomy identifies six *domains*—irreducible subject areas, each answering a single foundational question—plus a *metatheory* layer that records how the domains interact.

Ch.	Domain	Governing Question
1	BST	What naive set-theoretic objects does the metalanguage use?
2	SYN	How are well-formed expressions constructed?
3	SEM	How is meaning assigned to expressions?
4	DED	How are truths derived from assumptions?
5	CMP	What is effectively computable?
6	META	What are the properties of formal systems?
7	SET	What is the formal mathematical universe?
8	EXT	Where does the theory extend?

The domains are not independent. Their dependencies form the following graph:



Solid arrows indicate definitional dependencies: SYN uses BST, SEM uses SYN, and so on. Dashed arrows indicate *composition patterns*—metatheorems that live at the intersection of two or more domains. For instance, the Completeness Theorem (Chapter 6) connects SEM and DED but belongs to neither; it is a composition pattern.

An important architectural feature is the two-level treatment of set theory:

- **BST (Level 0)** provides the naive set-theoretic metalanguage—sets, functions, relations, cardinality—used to *build* the formal apparatus of logic.

- **SET (Level 1)** is Zermelo–Fraenkel set theory with Choice, treated as a formal first-order theory *within* the apparatus. Its axioms are sentences of $\mathcal{L}_\in$, and its theorems are derived using the proof systems of Chapter 4.

There is no circularity: Level-0 sets *build* the system; Level-1 set theory is a *subject of* the system.

The Taxonomy in Numbers

The taxonomy catalogs **205 formal items** across the six domains: 74 primitives (concepts taken as given within their domain), 60 definitions (concepts defined from primitives), 9 axioms (in formal set theory), and 62 theorems, lemmas, and propositions. These are connected by **13 composition patterns**—metatheorems such as Soundness (CP-001), Completeness (CP-002), Compactness (CP-003), and the Incompleteness Theorems (CP-005, CP-006)—that bridge domains.

Every formal item carries a unique identifier (e.g., `PRIM-SYN009` for the definition of a first-order language, `CP-002` for the Completeness Theorem). The full registry and dependency specifications are maintained in the `taxonomy/` directory of the project repository.

How to Read This Book

The chapters are arranged in dependency order: each chapter uses only material from earlier chapters. A linear reading from Chapter 1 through Chapter 7 encounters no forward references.

Three reading paths are natural:

1. **Foundations first** (Chapters 1–4, then 6): Set theory, syntax, semantics, deduction, completeness. This is the classical core of a first course in mathematical logic.

2. **Computability and incompleteness** (Chapters 1–2, 4–6): Syntax, deduction, computability, then the incompleteness theorems in Chapter 6. Semantics (Chapter 3) can be deferred.

3. **Set theory** (Chapters 1, 3–4, 6–7): The path to formal set theory, ordinals, and cardinals, using the metatheory of Chapter 6 as a bridge.

Chapter 8 (Extensions) is a brief guide to topics beyond the scope of this volume: modal logic, intuitionistic logic, many-valued logic, second-order logic, and the lambda calculus.

Relation to the Source Material

This text is compiled from the Open Logic Project's *Open Logic: A Complete Text*, released under a Creative Commons BY license. The systematization involved four phases:

1. **Taxonomy**: Identification of the six domains, their primitives, and the composition patterns connecting them.
2. **Mapping**: Every section of the source text was mapped to one or more taxonomy items, identifying redundancies and gaps.
3. **Compression**: Redundant treatments were merged, redundant proofs were cut or condensed, and new bridging material was written to unify the narrative.
4. **Recomposition**: The compressed material was assembled into the present volume, with uniform notation, consistent cross-referencing, and a single dependency-ordered structure.

The taxonomy, mapping, compression plan, and all quality-control audits are documented in the `taxonomy/` directory of the project repository at `github.com/sqkim25/OpenLog`

Acknowledgments

This volume would not exist without the Open Logic Project and its contributors. The original text, authored and maintained by Richard Zach and collaborators, represents years of careful exposition. We are grateful for their decision to release it as open-source educational material.

Contents

Chapter 1

Set-Theoretic Foundations

1.1 Sets and Membership

A *set* is a collection of objects, considered as a single object. The objects making up the set are called *elements* or *members* of the set. If x is an element of a set A, we write $x \in A$; if not, we write $x \notin A$. The set which has no elements is called the *empty* set and denoted "$\emptyset$".

It does not matter how we *specify* the set, or how we *order* its elements, or indeed how *many times* we count its elements. All that matters are what its elements are. We codify this in the following principle.

Definition 1.1 (Extensionality)**.** If A and B are sets, then $A = B$ iff every element of A is also an element of B, and vice versa.

Extensionality licenses some notation. In general, when we have some objects $a_1, \dots, a_n$, then $\{a_1, \dots, a_n\}$ is *the* set whose elements are $a_1, \dots, a_n$. We emphasise the word "*the*", since extensionality tells us that there can be only *one* such set. Indeed, extensionality also licenses the following:

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

This delivers on the point that, when we consider sets, we don't care about the order of their elements, or how many times they are specified.

Frequently we'll specify a set by some property that its elements share. We'll use the following shorthand notation for that: $\{x : \varphi(x)\}$, where the $\varphi(x)$ stands for the property that x has to have in order to be counted among the elements of the set.[1]

[1]Unrestricted set-builder notation leads to paradoxes: for instance, the set $R = \{x : x \notin x\}$ cannot exist (Russell's Paradox). Throughout this text, we work in naive set theory and rely on set-builder notation only for unproblematic instances of comprehension.

Extensionality gives us a way for showing that sets are identical: to show that $A = B$, show that whenever $x \in A$ then also $x \in B$, and whenever $y \in B$ then also $y \in A$.

We will often want to compare sets. One obvious kind of comparison is: *everything in one set is in the other too*.

Definition 1.2 (Subset)**.** If every element of a set A is also an element of B, then we say that A is a *subset* of B, and write $A \subseteq B$. If A is not a subset of B we write $A \not\subseteq B$. If $A \subseteq B$ but $A \neq B$, we write $A \subsetneq B$ and say that A is a *proper subset* of B.

The notation $(\forall x \in A)\varphi$ abbreviates $\forall x(x \in A \to \varphi)$, and $(\exists x \in A)\varphi$ abbreviates $\exists x(x \in A \land \varphi)$. Using this notation, $A \subseteq B$ iff $(\forall x \in A)\, x \in B$.

Proposition 1.3. *$A = B$ iff both $A \subseteq B$ and $B \subseteq A$.*

Now we consider a certain kind of set: the set of all subsets of a given set.

Definition 1.4 (Power Set)**.** The set consisting of all subsets of a set A is called the *power set of* A, written $\wp(A)$.

$$\wp(A) = \{B : B \subseteq A\}$$

Example 1.5. What are all the possible subsets of $\{a, b, c\}$? They are: $\emptyset$, $\{a\}$, $\{b\}$, $\{c\}$, $\{a, b\}$, $\{a, c\}$, $\{b, c\}$, $\{a, b, c\}$. The set of all these subsets is $\wp(\{a, b, c\})$:

$$\wp(\{a, b, c\}) = \{\emptyset, \{a\}, \{b\}, \{c\}, \{a, b\}, \{b, c\}, \{a, c\}, \{a, b, c\}\}$$

We can define new sets from old by combining their elements, or by taking only those elements they share.

Definition 1.6 (Union)**.** The *union* of two sets A and B, written $A \cup B$, is the set of all things which are elements of A, B, or both.

$$A \cup B = \{x : x \in A \lor x \in B\}$$

Definition 1.7 (Intersection)**.** The *intersection* of two sets A and B, written $A \cap B$, is the set of all things which are elements of both A and B.

$$A \cap B = \{x : x \in A \land x \in B\}$$

Two sets are called *disjoint* if their intersection is empty. This means they have no elements in common.

We can also form the union or intersection of more than two sets. An elegant way of dealing with this in general is the following: suppose you collect all the sets you want to form the union (or intersection) of into a single set. Then we can define the union of all our original sets as the set of all objects which belong to at least one element of the set, and the intersection as the set of all objects which belong to every element of the set.

Definition 1.8 (General Union)**.** If A is a set of sets, then $\bigcup A$ is the set of elements of elements of A:

$$\begin{aligned}\bigcup A &= \{x : x \text{ belongs to an element of } A\}\\ &= \{x : \text{there is a } B \in A \text{ so that } x \in B\}\end{aligned}$$

Definition 1.9 (General Intersection)**.** If A is a set of sets, then $\bigcap A$ is the set of objects which all elements of A have in common:

$$\begin{aligned}\bigcap A &= \{x : x \text{ belongs to every element of } A\}\\ &= \{x : \text{for all } B \in A, x \in B\}\end{aligned}$$

Example 1.10. Suppose $A = \{\{a, b\}, \{a, d, e\}, \{a, d\}\}$. Then $\bigcup A = \{a, b, d, e\}$ and $\bigcap A = \{a\}$.

We could also do the same for a sequence of sets A_1, A_2, ...

$$\begin{aligned}\bigcup_i A_i &= \{x : x \text{ belongs to one of the } A_i\}\\ \bigcap_i A_i &= \{x : x \text{ belongs to every } A_i\}.\end{aligned}$$

When we have an *index* of sets, i.e., some set I such that we are considering A_i for each $i \in I$, we may also write:

$$\begin{aligned}\bigcup_{i \in I} A_i &= \bigcup \{A_i : i \in I\}\\ \bigcap_{i \in I} A_i &= \bigcap \{A_i : i \in I\}\end{aligned}$$

The *set difference* $A \setminus B$ is the set of all elements of A which are not also elements of B, i.e., $A \setminus B = \{x : x \in A \text{ and } x \notin B\}$.

It follows from extensionality that sets have no order to their elements. So if we want to represent order, we use *ordered pairs* $\langle x, y\rangle$. In an unordered pair $\{x, y\}$, the order does not matter: $\{x, y\} = \{y, x\}$. In an ordered pair, it does: if $x \neq y$, then $\langle x, y\rangle \neq \langle y, x\rangle$.

We want to preserve the idea that ordered pairs are identical iff they share the same first element and share the same second element, i.e.:

$$\langle a, b\rangle = \langle c, d\rangle \text{ iff both } a = c \text{ and } b = d.$$

We can define ordered pairs in set theory using the Wiener–Kuratowski definition.

Definition 1.11 (Ordered pair)**.** $\langle a, b\rangle = \{\{a\}, \{a, b\}\}$.

This reduces ordered pairs to sets: $\langle a, b\rangle$ is simply the set $\{\{a\}, \{a, b\}\}$.

We can use ordered pairs to define ordered sequences of more than two objects. *Triples* $\langle x, y, z\rangle$ are $\langle\langle x, y\rangle, z\rangle$, *quadruples* $\langle x, y, z, u\rangle$ are $\langle\langle\langle x, y\rangle, z\rangle, u\rangle$, and so on. In general, we talk of *ordered n-tuples* $\langle x_1, \dots, x_n\rangle$.

Definition 1.12 (Cartesian product)**.** Given sets A and B, their *Cartesian product* $A \times B$ is defined by

$$A \times B = \{\langle x, y\rangle : x \in A \text{ and } y \in B\}.$$

If A is a set, the product of A with itself, $A \times A$, is also written A^2. It is the set of *all* pairs $\langle x, y\rangle$ with $x, y \in A$. The set of all triples $\langle x, y, z\rangle$ is A^3, and so on. We can give a recursive definition:

$$\begin{aligned} A^1 &= A \\ A^{k+1} &= A^k \times A \end{aligned}$$

Finite sequences (words) over A and the set A^* are defined formally in §1.4.

Definition 1.13 (Natural Numbers)**.** The set of *natural numbers* is denoted $\mathbb{N} = \{0, 1, 2, 3, \dots\}$.

The natural numbers form the most basic infinite set and serve as the foundation for counting, induction, and recursion throughout mathematical logic.

1.2 Relations

Recall from §1.1 the notion of an ordered pair $\langle a, b\rangle$ and the Cartesian product $A \times B$. We can use these to give a set-theoretic treatment of relations: a relation on a set is simply a set of ordered pairs.

Consider a particular relation on a set: the $<$-relation on the set $\mathbb{N}$ of natural numbers. Consider the set of all pairs of numbers $\langle n, m\rangle$ where $n < m$, i.e.,

$$R = \{\langle n, m\rangle : n, m \in \mathbb{N} \text{ and } n < m\}.$$

There is a close connection between n being less than m, and the pair $\langle n, m\rangle$ being a member of R, namely: $n < m$ iff $\langle n, m\rangle \in R$. Indeed, without any loss of information, we can consider the set R to *be* the $<$-relation on $\mathbb{N}$. This justifies the following definition:

Definition 1.14 (Binary relation)**.** A *binary relation* on a set A is a subset of A^2. If $R \subseteq A^2$ is a binary relation on A and $x, y \in A$, we sometimes write Rxy (or xRy) for $\langle x, y\rangle \in R$.

Example 1.15. The set $\mathbb{N}^2$ of pairs of natural numbers can be listed in a 2-dimensional matrix like this:

$$\begin{array}{ccccc}
\langle \mathbf{0}, \mathbf{0}\rangle & \langle 0, 1\rangle & \langle 0, 2\rangle & \langle 0, 3\rangle & \dots \\
\langle 1, 0\rangle & \langle \mathbf{1}, \mathbf{1}\rangle & \langle 1, 2\rangle & \langle 1, 3\rangle & \dots \\
\langle 2, 0\rangle & \langle 2, 1\rangle & \langle \mathbf{2}, \mathbf{2}\rangle & \langle 2, 3\rangle & \dots \\
\langle 3, 0\rangle & \langle 3, 1\rangle & \langle 3, 2\rangle & \langle \mathbf{3}, \mathbf{3}\rangle & \dots \\
\vdots & \vdots & \vdots & \vdots & \ddots
\end{array}$$

The diagonal pairs $\{\langle 0, 0\rangle, \langle 1, 1\rangle, \langle 2, 2\rangle, \dots\}$ form the *identity relation on* $\mathbb{N}$. We define $\mathrm{Id}_A = \{\langle x, x\rangle : x \in A\}$ for any set A. The pairs above the diagonal form the *less than* relation, those below the diagonal form the *greater than* relation.

According to this definition, *any* subset of A^2 is a relation on A. In particular, $\emptyset$ is a relation on any set (the empty relation), and A^2 itself is a relation on A (the universal relation).

Some kinds of relations are so common that they have been given special names. We categorize relations according to special properties. Combinations of these properties yield orders and equivalence relations.

Definition 1.16 (Reflexivity)**.** A relation $R \subseteq A^2$ is *reflexive* iff, for every $x \in A$, Rxx.

Definition 1.17 (Transitivity)**.** A relation $R \subseteq A^2$ is *transitive* iff, whenever Rxy and Ryz, then also Rxz.

Definition 1.18 (Symmetry)**.** A relation $R \subseteq A^2$ is *symmetric* iff, whenever Rxy, then also Ryx.

Definition 1.19 (Anti-symmetry)**.** A relation $R \subseteq A^2$ is *anti-symmetric* iff, whenever both Rxy and Ryx, then $x = y$ (or, in other words: if $x \neq y$ then either $\neg Rxy$ or $\neg Ryx$).

Note that being anti-symmetric and merely not being symmetric are different conditions. A relation can be both symmetric and anti-symmetric (e.g., the identity relation).

Definition 1.20 (Connectivity)**.** A relation $R \subseteq A^2$ is *connected* if for all $x, y \in A$, if $x \neq y$, then either Rxy or Ryx.

Definition 1.21 (Irreflexivity)**.** A relation $R \subseteq A^2$ is called *irreflexive* if, for all $x \in A$, not Rxx.

Definition 1.22 (Asymmetry)**.** A relation $R \subseteq A^2$ is called *asymmetric* if for no pair $x, y \in A$ we have both Rxy and Ryx.

Note that if $A \neq \emptyset$, then no irreflexive relation on A is reflexive and every asymmetric relation on A is also anti-symmetric. However, there are $R \subseteq A^2$ that are not reflexive and also not irreflexive, and there are anti-symmetric relations that are not asymmetric.

The identity relation on a set is reflexive, symmetric, and transitive. Relations R that have all three of these properties are very common.

Definition 1.23 (Equivalence relation)**.** A relation $R \subseteq A^2$ that is reflexive, symmetric, and transitive is called an *equivalence relation*. Elements x and y

of A are said to be *R-equivalent* if Rxy.

Equivalence relations give rise to the notion of an *equivalence class*. An equivalence relation "chunks up" the domain into different partitions. Within each partition, all the objects are related to one another; and no objects from different partitions relate to one another.

Definition 1.24 (Equivalence class)**.** Let $R \subseteq A^2$ be an equivalence relation. For each $x \in A$, the *equivalence class* of x in A is the set $[x]_R = \{y \in A : Rxy\}$. The *quotient* of A under R is $A/_R = \{[x]_R : x \in A\}$, i.e., the set of these equivalence classes.

The next result proves that the equivalence classes are indeed the partitions of A:

Proposition 1.25. *If $R \subseteq A^2$ is an equivalence relation, then Rxy iff $[x]_R = [y]_R$.*

Proof. For the left-to-right direction, suppose Rxy, and let $z \in [x]_R$. By definition, then, Rxz. Since R is an equivalence relation, Ryz. (Spelling this out: as Rxy and R is symmetric we have Ryx, and as Rxz and R is transitive we have Ryz.) So $z \in [y]_R$. Generalising, $[x]_R \subseteq [y]_R$. But exactly similarly, $[y]_R \subseteq [x]_R$. So $[x]_R = [y]_R$, by extensionality.

For the right-to-left direction, suppose $[x]_R = [y]_R$. Since R is reflexive, Ryy, so $y \in [y]_R$. Thus also $y \in [x]_R$ by the assumption that $[x]_R = [y]_R$. So Rxy. □

Example 1.26. A nice example of equivalence relations comes from modular arithmetic. For any a, b, and $n \in \mathbb{Z}^+$, say that $a \equiv_n b$ iff dividing a by n gives the same remainder as dividing b by n. (Somewhat more symbolically: $a \equiv_n b$ iff, for some $k \in \mathbb{Z}$, $a - b = kn$.) Now, $\equiv_n$ is an equivalence relation, for any n. And there are exactly n distinct equivalence classes generated by $\equiv_n$; that is, $\mathbb{N}/_{\equiv_n}$ has n elements. These are: the set of numbers divisible by n without remainder, i.e., $[0]_{\equiv_n}$; the set of numbers divisible by n with remainder 1, i.e., $[1]_{\equiv_n}$; ...; and the set of numbers divisible by n with remainder $n-1$, i.e., $[n-1]_{\equiv_n}$.

Many of our comparisons involve describing some objects as being "less than", "equal to", or "greater than" other objects, in a certain respect. These involve *order* relations. There are different kinds: some require that any two objects be comparable, others don't; some include identity (like $\leq$) and some exclude it (like $<$).

Definition 1.27 (Preorder)**.** A relation which is both reflexive and transitive is called a *preorder.*

Definition 1.28 (Partial order)**.** A preorder which is also anti-symmetric is called a *partial order*.

Definition 1.29 (Linear order)**.** A partial order which is also connected is called a *total order* or *linear order.*

Definition 1.30 (Strict order)**.** A *strict order* is a relation which is irreflexive, asymmetric, and transitive.

Definition 1.31 (Strict linear order)**.** A strict order which is also connected is called a *strict total order* or *strict linear order.*

Example 1.32. An important partial order is the relation $\subseteq$ on a set of sets. This is not in general a linear order, since if $a \neq b$ and we consider $\wp(\{a, b\}) = \{\emptyset, \{a\}, \{b\}, \{a, b\}\}$, we see that $\{a\} \nsubseteq \{b\}$ and $\{a\} \neq \{b\}$ and $\{b\} \nsubseteq \{a\}$.

Any strict order R on A can be turned into a partial order by adding the diagonal Id_A, i.e., adding all the pairs $\langle x, x \rangle$. (This is called the *reflexive closure* of R.)

Proposition 1.33. *If R is a strict order on A, then $R^+ = R \cup \mathrm{Id}_A$ is a partial order. Moreover, if R is a strict linear order, then R^+ is a linear order.*

Proof sketch. Suppose R is a strict order on A and let $R^+ = R \cup \mathrm{Id}_A$. *Reflexivity*: $\langle x, x \rangle \in \mathrm{Id}_A \subseteq R^+$ for all $x \in A$. *Anti-symmetry*: if R^+xy and R^+yx with $x \neq y$, then Rxy and Ryx, contradicting asymmetry. *Transitivity*: if R^+xy and R^+yz, then either both pairs are in R (use transitivity of R), or one is in Id_A (use $x = y$ or $y = z$). The "moreover" clause follows because connectivity of R is inherited by R^+. □

A particular kind of partial order that plays an important role in logic is a *tree*. Finite trees occur in syntax (formula decomposition) and derivation systems, while infinite trees appear in completeness proofs.

A *minimal element* in a set A partially ordered by $\leq$ is an element $x \in A$ such that for all $y \in A$ we have that $x \leq y$. A set is *well-ordered* by $\leq$ if every one of its subsets has a minimal element.

Definition 1.34 (Tree)**.** A *tree* is a pair $T = \langle A, \leq \rangle$ such that A is a set and $\leq$ is a partial order on A with a unique minimal element $r \in A$ (called the *root*) such that for all $x \in A$, the set $\{y : y \leq x\}$ is well-ordered by $\leq$.

Definition 1.35 (Successors)**.** Suppose $T = \langle A, \leq \rangle$ is a tree. If $x, y \in A$, $x < y$, and there is no $z \in A$ such that $x < z < y$, then we say that y is a *successor* (or *child*) of x, and x is the *predecessor* (or *parent*) of y.

Definition 1.36 (Branches)**.** Given a tree $T = \langle A, \leq \rangle$, a *branch* of T is a maximal chain in T, i.e., a set $B \subseteq A$ such that for any $x, y \in B$ either $x \leq y$ or $y \leq x$, and for any $z \in A \setminus B$ there exists $u \in B$ such that neither $z \leq u$ nor $u \leq z$. We use $[T]$ to denote the set of all branches of T.

A tree is *infinite* if its underlying set is infinite, and *finitely branching* if every node has only finitely many successors.

Example 1.37. A classic example of a finitely branching tree is the *infinite binary tree* $\mathbb{B}^*$, ordered by the extension relation $\sqsubseteq$ (e.g., $101 \sqsubseteq 101101$). Every element s has exactly two successors, $s0$ and $s1$, and its root is the empty sequence Λ.

It is often useful to modify or combine relations. Here are some fundamental operations.

Definition 1.38 (Operations on relations)**.** Let R, S be relations, and A be any set.

The *inverse* of R is $R^{-1} = \{\langle y, x \rangle : \langle x, y \rangle \in R\}$.
The *relative product* of R and S is $(R \mid S) = \{\langle x, z \rangle : \exists y (Rxy \wedge Syz)\}$.
The *restriction* of R to A is $R{\restriction}_A = R \cap A^2$.
The *application* of R to A is $R[A] = \{y : (\exists x \in A)\, Rxy\}$.

Definition 1.39 (Transitive closure)**.** Let $R \subseteq A^2$ be a binary relation.

The *transitive closure* of R is $R^+ = \bigcup_{0<n\in\mathbb{N}} R^n$, where we recursively define $R^1 = R$ and $R^{n+1} = R^n \mid R$.

The *reflexive transitive closure* of R is $R^* = R^+ \cup \mathrm{Id}_A$.

1.3 Functions

A *function* is a map which sends each element of a given set to a specific element in some (other) given set. For instance, the operation of adding 1 defines a function: each number n is mapped to a unique number $n+1$.

More generally, functions may take pairs, triples, etc., as inputs and return some kind of output. In a mathematical, abstract sense, a function is a *black box*: what matters is only what output is paired with what input, not the method for calculating the output.

Definition 1.40 (Function)**.** A *function* $f\colon A \to B$ is a mapping of each element of A to an element of B.

We call A the *domain* of f and B the *codomain* of f. The elements of A are called inputs or *arguments* of f, and the element of B that is paired with an argument x by f is called the *value of* f for argument x, written $f(x)$.

The *range* $\mathrm{ran}(f)$ of f is the subset of the codomain consisting of the values of f for some argument; $\mathrm{ran}(f) = \{f(x) : x \in A\}$.

Example 1.41. Let $f\colon \mathbb{N} \to \mathbb{N}$ be defined such that $f(x) = x + 1$. This is a function which takes in natural numbers and outputs natural numbers. Given a natural number x, f will output its successor $x+1$. In this case, the codomain $\mathbb{N}$ is not the range of f, since the natural number 0 is not the successor of any natural number. The range of f is the set of all positive integers, $\mathbb{Z}^+$.

Two functions f and g are the same if they have the same domain, codomain, and agree on all values: if $\forall x\, f(x) = g(x)$, then $f = g$.

We introduce a taxonomy for the most frequently encountered kinds of functions.

Definition 1.42 (Surjective function)**.** A function $f\colon A \to B$ is *surjective* iff B is also the range of f, i.e., for every $y \in B$ there is at least one $x \in A$ such that $f(x) = y$, or in symbols:

$$(\forall y \in B)(\exists x \in A)\, f(x) = y.$$

We call such a function a surjection from A to B.

Definition 1.43 (Injective function)**.** A function $f\colon A \to B$ is *injective* iff for each $y \in B$ there is at most one $x \in A$ such that $f(x) = y$. We call such a function an injection from A to B.

Definition 1.44 (Bijection)**.** A function $f\colon A \to B$ is *bijective* iff it is both surjective and injective. We call such a function a bijection from A to B (or between A and B).

Example 1.45. The constant function $f\colon \mathbb{N} \to \mathbb{N}$ given by $f(x) = 1$ is neither injective, nor surjective. The identity function $f\colon \mathbb{N} \to \mathbb{N}$ given by $f(x) = x$ is both injective and surjective. The successor function $f\colon \mathbb{N} \to \mathbb{N}$ given by $f(x) = x + 1$ is injective but not surjective. The function $f\colon \mathbb{N} \to \mathbb{N}$ defined by:

$$f(x) = \begin{cases} \frac{x}{2} & \text{if } x \text{ is even} \\ \frac{x+1}{2} & \text{if } x \text{ is odd.} \end{cases}$$

is surjective, but not injective.

We now ask whether the mapping defined by a function can be "reversed." This is made precise by the notion of an inverse.

Definition 1.46 (Inverse)**.** A function $g\colon B \to A$ is an *inverse* of a function $f\colon A \to B$ if $f(g(y)) = y$ and $g(f(x)) = x$ for all $x \in A$ and $y \in B$.

If f has an inverse g, we often write f^{-1} instead of g.

Proposition 1.47. *If $f\colon A \to B$ is injective, then there is a* left inverse $g\colon B \to A$ *of f so that $g(f(x)) = x$ for all $x \in A$.*

Proof sketch. Since f is injective, for each $y \in \operatorname{ran}(f)$ there is exactly one x with $f(x) = y$; define $g(y) = x$ in that case. For $y \notin \operatorname{ran}(f)$, map $g(y)$ to any fixed $a \in A$. Then $g(f(x)) = x$ for all $x \in A$ by construction. $\square$

Proposition 1.48. *If $f\colon A \to B$ is surjective, then there is a* right inverse $h\colon B \to A$ *of f so that $f(h(y)) = y$ for all $y \in B$.*

The proof requires choosing, for each $y \in B$, some $x \in A$ with $f(x) = y$. That such a choice is always possible is guaranteed by the Axiom of Choice. In many specific cases (e.g., when $A = \mathbb{N}$, or when A is finite, or when f is bijective), the Axiom of Choice is not required.

Proposition 1.49. *If $f\colon A \to B$ is bijective, there is a function $f^{-1}\colon B \to A$ so that for all $x \in A$, $f^{-1}(f(x)) = x$ and for all $y \in B$, $f(f^{-1}(y)) = y$.*

Proof sketch. Since f is injective, it has a left inverse g. Since f is surjective, it has a right inverse h. For any $y \in B$: $g(y) = g(f(h(y))) = h(y)$, so $g = h$. This common function $f^{-1} = g = h$ is both a left and right inverse of f. □

We can define a new function by composing two functions f and g, i.e., by first applying f and then g. This is only possible if the range of f is a subset of the domain of g.

Definition 1.50 (Composition)**.** Let $f\colon A \to B$ and $g\colon B \to C$ be functions. The *composition* of f with g is $g \circ f\colon A \to C$, where $(g \circ f)(x) = g(f(x))$.

Example 1.51. Consider the functions $f(x) = x + 1$, and $g(x) = 2x$. Since $(g \circ f)(x) = g(f(x))$, for each input x you must first take its successor, then multiply the result by two. So their composition is given by $(g \circ f)(x) = 2(x + 1)$.

Composition preserves injectivity and surjectivity: if $f\colon A \to B$ and $g\colon B \to C$ are both injective, then $g \circ f$ is injective; likewise for surjectivity.

A function $f\colon A \to B$ naturally defines a relation between A and B. In fact, we can *identify* a function with this relation, i.e., with a set of pairs.

Definition 1.52 (Graph of a function)**.** Let $f\colon A \to B$ be a function. The *graph* of f is the relation $R_f \subseteq A \times B$ defined by

$$R_f = \{\langle x, y\rangle : f(x) = y\}.$$

Conversely, any relation $R \subseteq A \times B$ that is functional (for each $x \in A$ there is exactly one $y \in B$ with Rxy) is the graph of a function $f\colon A \to B$ defined by $f(x) = y$ iff Rxy. Functions can thus be identified with certain relations, i.e., with certain sets of tuples.

1.4 Sequences and Numbers

The ordered pair $\langle a, b\rangle$ (PRIM-BST006, §1.1) and Cartesian product $A \times B$ (PRIM-BST007, §1.1) allow us to form pairs and grids of elements. We now extend these constructions to *finite sequences* of arbitrary length and to *infinite sequences*, both of which are pervasive in logic and computability theory.

1.4.1 Tuples and Finite Sequences

Ordered pairs generalise to longer sequences by iteration. A *triple* $\langle x, y, z\rangle$ is the pair $\langle\langle x, y\rangle, z\rangle$; a *quadruple* $\langle x, y, z, u\rangle$ is $\langle\langle\langle x, y\rangle, z\rangle, u\rangle$; and in general an *ordered n-tuple* $\langle x_1, \ldots, x_n\rangle$ is defined recursively by

$$\langle x_1\rangle = x_1,$$
$$\langle x_1, \ldots, x_{k+1}\rangle = \langle\langle x_1, \ldots, x_k\rangle, x_{k+1}\rangle.$$

The Cartesian product generalises correspondingly:

$$A^1 = A,$$
$$A^{k+1} = A^k \times A.$$

Definition 1.53 (Finite sequences / words)**.** Let A be a set. A *word* (or *finite sequence*) over A is any n-tuple of elements of A, for some $n \geq 0$. By convention elements of A are sequences of length 1, and $\emptyset$ is the unique sequence of length 0 (the *empty word*). The set of *all* finite sequences over A is

$$A^* = \{\emptyset\} \cup A \cup A^2 \cup A^3 \cup \ldots$$

Example 1.54. If $A = \{a, b, c\}$, then the sequence "*bac*" is the triple $\langle b, a, c\rangle \in A^3$. The set A^* contains the empty word, all single letters, all pairs, all triples, and so on.

1.4.2 Infinite Sequences

Definition 1.55 (Infinite sequences)**.** For any set A, the set A^ω consists of all infinite (one-way) sequences of elements of A. An element of A^ω is a sequence $a_1a_2a_3\ldots$ where each $a_i \in A$. Equivalently, A^ω is the set of all functions $f\colon \mathbb{N} \to A$ (or $f\colon \mathbb{Z}^+ \to A$, depending on indexing convention).

Finite sequences (A^*, PRIM-BST010) and infinite sequences (A^ω, PRIM-BST011) will recur throughout the text: A^* underlies the syntax of formal languages (see PRIM-SYN001, CH-SYN), while A^ω appears in diagonalisation arguments (§1.6).

With the machinery of sequences and numbers in place, we now develop the proof methods and recursive definitions that operate on these structures.

1.5 Induction and Recursion

Induction is a proof technique for establishing that *every* element of a suitably constructed set has a given property. It applies whenever the set is built up from basic elements by repeatedly applying certain operations—that is, whenever the set is the *closure* of some base elements under those operations.

A set S is *closed* under a function f iff $x \in S$ implies $f(x) \in S$. A property P is *preserved* under f iff $P(a)$ implies $P(f(a))$ for every a in the domain of f.

1.5.1 Mathematical Induction on $\mathbb{N}$

The natural numbers $\mathbb{N}$ (PRIM-BST012, §1.1) are closed under the successor function $s(n) = n + 1$, and every natural number is obtained from 0 by finitely many applications of s. This yields the principle of mathematical induction.

Definition 1.56 (Mathematical induction on $\mathbb{N}$)**.** If 0 has property P, and P is preserved under the successor function, then every natural number has property P.

More formally: if $P(0)$ holds and $P(k) \Rightarrow P(k+1)$ for all $k \in \mathbb{N}$, then $P(n)$ holds for all $n \in \mathbb{N}$.

Remark 1 (Set-theoretic justification). Induction on $\mathbb{N}$ can be grounded in the set-theoretic notion of closure. If N, s, o form a Dedekind algebra (see below), then for any set X: if $o \in X$ and $s(n) \in X$ whenever $n \in N \cap X$, then $N \subseteq X$. Applying this with $X = \{n \in N : P(n)\}$ recovers the familiar induction schema.

1.5.2 Structural Induction on Formulas

The set of formulas of a formal language (see §2.2 for the formal definition) is built from atomic formulas by the formula-building functions: negation ($\neg$), conjunction ($\wedge$), disjunction ($\vee$), conditional ($\rightarrow$), biconditional ($\leftrightarrow$), and quantification ($\forall$, $\exists$). The set of formulas is closed under each of these operations, and every formula is obtained from atomic formulas by finitely many applications of them.

Definition 1.57 (Structural induction on formulas)**.** If every atomic formula has property P, and P is preserved under each formula-building function, then every formula has property P.

This suggests the following recipe for an inductive proof that every formula has property P:

Base case. Let φ be an atomic formula. [...] Therefore φ has property P.

Inductive step. Let φ and ψ be formulas, both having property P.

- Case $\neg$: [...] Therefore $\neg\varphi$ has property P.

- Case $\wedge$: [...] Therefore $\varphi \wedge \psi$ has property P.
- Case $\vee$: [...] Therefore $\varphi \vee \psi$ has property P.
- Case $\rightarrow$: [...] Therefore $\varphi \rightarrow \psi$ has property P.
- Case $\leftrightarrow$: [...] Therefore $\varphi \leftrightarrow \psi$ has property P.
- Case $\forall$: [...] Therefore $\forall x \varphi$ has property P.
- Case $\exists$: [...] Therefore $\exists x \varphi$ has property P.

Therefore every formula has property P.

1.5.3 Closure and Dedekind Algebras

The closure machinery underlying induction can be made precise in purely set-theoretic terms.

Definition 1.58 (Closure)**.** For any function f and any element o, a set X is f-*closed* iff $f(x) \in X$ for every $x \in X$. The *closure* of o under f is

$$\mathrm{clo}_f(o) \;=\; \bigcap\{X : o \in X \text{ and } X \text{ is } f\text{-closed}\}.$$

Intuitively, $\mathrm{clo}_f(o)$ is the *smallest* f-closed set containing o. One can verify that $o \in \mathrm{clo}_f(o)$, that $\mathrm{clo}_f(o)$ is f-closed, and that it is contained in every f-closed set that has o as an element.

Definition 1.59 (Dedekind algebra)**.** A *Dedekind algebra* is a triple (A, f, o) where A is a set, $f\colon A \to A$ is a function, and $o \in A$, satisfying:

1. $o \notin \mathrm{ran}(f)$ (zero is not a successor),
2. f is an injection (distinct elements have distinct successors),
3. $A = \mathrm{clo}_f(o)$ (A is the smallest f-closed set containing o).

Any Dedekind algebra can serve as a surrogate for the natural numbers: conditions (1)–(3) are precisely the Dedekind–Peano characterisation of $\mathbb{N}$, and arithmetic (addition, multiplication, exponentiation) can be defined by recursion on a Dedekind algebra. Moreover, any Dedekind infinite set (see DEF-BST008, §1.6) gives rise to a Dedekind algebra.

1.6 Cardinality

With functions (PRIM-BST009), injections (DEF-BST001), surjections (DEF-BST002), and bijections (DEF-BST003) in hand (§1.3), we can make precise the idea of "how many elements" a set has, and compare the sizes of infinite sets.

1.6.1 Enumerations and Enumerable Sets

Informally, an *enumeration* of a set A is a (possibly infinite) list of elements of A such that every element appears at some finite position. We make this precise as follows.

Definition 1.60 (Enumeration). An *enumeration* of a non-empty set A is a surjective function $f\colon \mathbb{N} \to A$ (equivalently, $f\colon \mathbb{Z}^+ \to A$).

A surjection $f\colon \mathbb{Z}^+ \to A$ gives the list $f(1), f(2), f(3), \ldots$ in which every element of A appears. The list may contain repetitions; these can always be removed (by skipping any $f(k)$ already listed), yielding a bijection between A and either $\mathbb{N}$ or an initial segment $\{0, \ldots, n\}$ (equivalently, $\mathbb{Z}^+$ or $\{1, \ldots, n\}$), depending on whether A is infinite or finite.

Definition 1.61 (Enumerable set). A set A is *enumerable* (also called *countable*) iff $A = \emptyset$ or there is an enumeration of A. A set is *non-enumerable* iff it is not enumerable.

Corollary 1.62 ($\mathbb{N}$ is enumerable). *$\mathbb{N}$ is enumerable.*

Proof. The identity function $\mathrm{Id}_{\mathbb{N}}(n) = n$ is a bijection $\mathbb{N} \to \mathbb{N}$, hence an enumeration of $\mathbb{N}$. □

Example 1.63 (Enumerating $\mathbb{Z}$ by hopping). The function $f\colon \mathbb{N} \to \mathbb{Z}$ defined by

$$f(n) = (-1)^n \left\lceil \tfrac{n}{2} \right\rceil$$

enumerates the integers by "hopping" between positive and negative values:

$f(0)$	$f(1)$	$f(2)$	$f(3)$	$f(4)$	$f(5)$	$f(6)$	$\ldots$
0	-1	1	-2	2	-3	3	$\ldots$

Equivalently, f can be written as:

$$f(n) = \begin{cases} n/2 & \text{if } n \text{ is even,} \\ -(n+1)/2 & \text{if } n \text{ is odd.} \end{cases}$$

1.6.2 Cantor's Zig-Zag Method

We now show that the set of all pairs of natural numbers is enumerable. Arrange the elements of $\mathbb{N} \times \mathbb{N}$ in an array:

	0	**1**	**2**	**3**	...
0	$\langle 0,0\rangle$	$\langle 0,1\rangle$	$\langle 0,2\rangle$	$\langle 0,3\rangle$	...
1	$\langle 1,0\rangle$	$\langle 1,1\rangle$	$\langle 1,2\rangle$	$\langle 1,3\rangle$	...
2	$\langle 2,0\rangle$	$\langle 2,1\rangle$	$\langle 2,2\rangle$	$\langle 2,3\rangle$	...
3	$\langle 3,0\rangle$	$\langle 3,1\rangle$	$\langle 3,2\rangle$	$\langle 3,3\rangle$	...
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$

Every ordered pair appears exactly once. To convert this two-dimensional array into a one-dimensional list, traverse the successive anti-diagonals (where $n + m$ is constant), reading each anti-diagonal from bottom-left to top-right. Numbering positions from 0, we obtain:

	0	**1**	**2**	**3**	**4**	...
0	0	1	3	6	10	...
1	2	4	7	11	...	...
2	5	8	12	...	...	...
3	9	13	...	...	...	...
4	14	...	...	...	...	...
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	...	$\ddots$

This pattern is called *Cantor's zig-zag method*. It enumerates $\mathbb{N} \times \mathbb{N}$ as:

$$\langle 0,0\rangle,\ \langle 0,1\rangle,\ \langle 1,0\rangle,\ \langle 0,2\rangle,\ \langle 1,1\rangle,\ \langle 2,0\rangle,\ \langle 0,3\rangle,\ \langle 1,2\rangle,\ \langle 2,1\rangle,\ \langle 3,0\rangle,\ \ldots$$

Proposition 1.64 ($\mathbb{N} \times \mathbb{N}$ is enumerable)**.** $\mathbb{N} \times \mathbb{N}$ *is enumerable.*

Proof. Let $f\colon \mathbb{N} \to \mathbb{N} \times \mathbb{N}$ map each $k \in \mathbb{N}$ to the pair $\langle n, m\rangle$ occupying position k in the zig-zag array. Every pair $\langle n, m\rangle$ lies on the anti-diagonal where $n + m$ is constant, and so appears at a definite finite position; hence f is a surjection (indeed a bijection). □

The same technique generalises by induction. We can view $\mathbb{N}^3 = (\mathbb{N} \times \mathbb{N}) \times \mathbb{N}$ and enumerate it by zig-zagging the enumeration of $\mathbb{N}^2$ against $\mathbb{N}$. Repeating:

Proposition 1.65. $\mathbb{N}^n$ *is enumerable, for every* $n \in \mathbb{N}$.

1.6.3 Pairing Functions

It is useful to have an explicit formula for the zig-zag enumeration. The *inverse* of the enumeration is a bijection $g\colon \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ that assigns to each pair its position in the zig-zag array.

Definition 1.66 (Pairing function)**.** An arithmetical *pairing function* is an injective function $f\colon A \times B \to \mathbb{N}$. The value $f(x, y)$ is called the *code* for $\langle x, y\rangle$.

The zig-zag order yields the explicit pairing function

$$g(n, m) = \frac{(n + m + 1)(n + m)}{2} + n.$$

Indeed, the pair $\langle n, m\rangle$ lies on the anti-diagonal where the sum $n + m$ equals some value k; the kth triangular number $k(k + 1)/2$ gives the position of the first entry on that anti-diagonal, and n counts the offset within it. For instance, $g(1, 2) = \frac{4\cdot 3}{2} + 1 = 7$, matching the array above.

1.6.4 Non-Enumerable Sets

Not every infinite set is enumerable. The *diagonal method*, introduced by Cantor, shows that certain natural sets have "more" elements than can be listed.

Theorem 1.67 ($\mathbb{B}^\omega$ is non-enumerable)**.** *$\mathbb{B}^\omega$, the set of all infinite sequences of* 0*'s and* 1*'s, is non-enumerable.*

Proof. Suppose for contradiction that $\mathbb{B}^\omega$ is enumerable. Then there is a list $s_0, s_1, s_2, \ldots$ of all elements of $\mathbb{B}^\omega$. Write $s_i(j)$ for the jth entry of the ith sequence. Arrange the sequences into an array:

	0	1	2	3	$\ldots$
0	$\mathbf{s_0(0)}$	$s_0(1)$	$s_0(2)$	$s_0(3)$	$\ldots$
1	$s_1(0)$	$\mathbf{s_1(1)}$	$s_1(2)$	$s_1(3)$	$\ldots$
2	$s_2(0)$	$s_2(1)$	$\mathbf{s_2(2)}$	$s_2(3)$	$\ldots$
3	$s_3(0)$	$s_3(1)$	$s_3(2)$	$\mathbf{s_3(3)}$	$\ldots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$

Define a new sequence $d \in \mathbb{B}^\omega$ by flipping each diagonal entry:

$$d(n) = \begin{cases} 1 & \text{if } s_n(n) = 0, \\ 0 & \text{if } s_n(n) = 1. \end{cases}$$

Then $d \in \mathbb{B}^\omega$, since it is an infinite sequence of 0's and 1's. But for every $n \in \mathbb{N}$, $d(n) \neq s_n(n)$, so $d \neq s_n$. Hence d does not appear on the list, contradicting the assumption that the list enumerates all of $\mathbb{B}^\omega$. $\square$

Theorem 1.68 ($\wp(\mathbb{N})$ is non-enumerable)**.** *$\wp(\mathbb{N})$ is not enumerable.*

Proof sketch. The argument follows the same diagonal pattern. Given any list $N_0, N_1, N_2, \ldots$ of subsets of $\mathbb{N}$, define $D = \{n \in \mathbb{N} : n \notin N_n\}$. Then $D \subseteq \mathbb{N}$ but $D \neq N_n$ for every n (since $n \in D$ iff $n \notin N_n$), so the list omits D. □

1.6.5 Reduction

Diagonalisation is not the only way to establish non-enumerability. A general strategy is *reduction*: to show that a set B is non-enumerable, exhibit a surjection $f \colon B \to A$ from B onto a known non-enumerable set A. Any enumeration of B would then yield an enumeration of A, a contradiction.

Equivalently, if A is non-enumerable and there is an injection $g \colon A \to B$, then B is non-enumerable.

1.6.6 Equinumerosity

To compare sizes of arbitrary sets—not just to classify them as "enumerable" or "non-enumerable"—we introduce a general notion of size equivalence.

Definition 1.69 (Equinumerosity)**.** A is *equinumerous* with B, written $A \approx B$, iff there is a bijection $f \colon A \to B$.

Equinumerosity is an equivalence relation: reflexivity follows from the identity function (DEF-BST003), symmetry from inverses of bijections, and transitivity from composition of bijections. Moreover, if $A \approx B$, then A is enumerable if and only if B is.

1.6.7 Dedekind Infinite Sets

Definition 1.70 (Dedekind infinite)**.** A set A is *Dedekind infinite* iff there is an injection from A to a proper subset of A. Equivalently, there exist some $o \in A$ and an injection $f \colon A \to A$ such that $o \notin \operatorname{ran}(f)$.

Intuitively, a Dedekind infinite set can be put in bijection with a proper subset of itself—the hallmark of infinity. As noted in §1.5, any Dedekind infinite set gives rise to a Dedekind algebra and hence supports induction.

1.6.8 Sets of Different Sizes and Cantor's Theorem

We now define a strict size ordering on sets and state the central theorem of cardinality theory.

Definition 1.71 (Size comparison by injection)**.** A is *no larger than* B, written $A \preceq B$, iff there is an injection $f\colon A \to B$.

Definition 1.72 (Strict size comparison)**.** A is *smaller than* B, written $A \prec B$, iff $A \preceq B$ and $A \not\approx B$—i.e., there is an injection $f\colon A \to B$ but no bijection $g\colon A \to B$.

The relation $\cdot \preceq \cdot$ is reflexive and transitive; the relation $\cdot \prec \cdot$ is irreflexive and transitive.

Using these definitions, a set A is enumerable iff $A \preceq \mathbb{N}$, and non-enumerable iff $\mathbb{N} \prec A$. The non-enumerability of $\wp(\mathbb{N})$ (Theorem 1.68) can be restated as $\mathbb{N} \prec \wp(\mathbb{N})$. Cantor proved that this phenomenon is perfectly general:

Theorem 1.73 (Cantor's Theorem)**.** *$A \prec \wp(A)$, for any set A.*

Proof. *$A \preceq \wp(A)$:* The function $f\colon A \to \wp(A)$ defined by $f(x) = \{x\}$ is an injection, since $x \neq y$ implies $\{x\} \neq \{y\}$ by extensionality (PRIM-BST001).

$A \not\approx \wp(A)$: Suppose for contradiction that there is a bijection $g\colon A \to \wp(A)$. Define

$$D = \{x \in A : x \notin g(x)\}.$$

Since $g(x) \subseteq A$ for every x, D is a well-defined subset of A, so $D \in \wp(A)$. Because g is a bijection, there exists $y \in A$ with $g(y) = D$. But then

$$y \in g(y) \text{ iff } y \in D \text{ iff } y \notin g(y),$$

a contradiction. Hence no bijection $A \to \wp(A)$ exists, and $A \not\approx \wp(A)$.

Combining both parts: $A \prec \wp(A)$. □

The construction of the "anti-diagonal" set D mirrors the diagonal argument in Theorem 1.68, and was the inspiration for Russell's Paradox.

1.6.9 Schröder-Bernstein

Cantor's Theorem shows that strict size differences exist among infinite sets. For the converse direction—showing two sets have the *same* size—the following deep result is essential.

Theorem 1.74 (Schröder-Bernstein). *If $A \preceq B$ and $B \preceq A$, then $A \approx B$.*

In other words, if there is an injection $f\colon A \to B$ and an injection $g\colon B \to A$, then there is a bijection $h\colon A \to B$. This justifies treating $\cdot \preceq \cdot$ as a genuine size comparison: mutual "no larger than" implies equinumerosity.

Proof sketch. Given injections $f\colon A \to B$ and $g\colon B \to A$, the composition $g \circ f$ is an injection from A into A whose range $g[f[A]]$ satisfies $g[f[A]] \subseteq g[B] \subseteq A$. Thus $g \circ f$ is a bijection from A onto $g[f[A]]$, and we have the "subset sandwich" $g[f[A]] \subseteq g[B] \subseteq A$ with $g[f[A]] \approx A$.

The key step uses the generalised closure construction (cf. DEF-BST006, §1.5). Define $\mathrm{Clo}_{(g\circ f)}(A \setminus g[B])$: the smallest $(g \circ f)$-closed set containing $A \setminus g[B]$. One shows that the function

$$h(x) = \begin{cases} (g \circ f)(x) & \text{if } x \text{ belongs to the closure,} \\ x & \text{otherwise} \end{cases}$$

is a bijection from A onto $g[B]$. Composing h with $g^{-1}\colon g[B] \to B$ (which exists since g is injective) yields a bijection $A \to B$. □

Chapter 2

Syntax

2.1 Languages and Symbols

Expressions of first-order logic are built up from a basic vocabulary containing *variables*, *constant symbols*, *predicate symbols*, and sometimes *function symbols*. From them, together with logical connectives, quantifiers, and punctuation symbols such as parentheses and commas, *terms* and *formulas* are formed.

Predicate symbols are names for properties and relations, constant symbols are names for individual objects, and function symbols are names for mappings. These, except for the identity predicate $=$, are the *non-logical symbols* and together make up a language. Any first-order language $\mathcal{L}$ is determined by its non-logical symbols. In the most general case, $\mathcal{L}$ contains infinitely many symbols of each kind.

In the general case, we make use of the following symbols in first-order logic:

1. **Logical symbols**

    a) Logical connectives: $\neg$ (negation), $\wedge$ (conjunction), $\vee$ (disjunction), $\rightarrow$ (conditional), $\leftrightarrow$ (biconditional), $\forall$ (universal quantifier), $\exists$ (existential quantifier).

    b) The propositional constant for falsity $\bot$.

    c) The propositional constant for truth $\top$.

    d) The two-place identity predicate $=$.

    e) A denumerable set of variables: v_0, v_1, v_2, …

2. **Non-logical symbols**, making up the *standard language* of first-order logic

    a) A denumerable set of n-place predicate symbols for each $n > 0$: A^n_0, A^n_1, A^n_2, …

b) A denumerable set of constant symbols: c_0, c_1, c_2, ...
c) A denumerable set of n-place function symbols for each $n > 0$: f_0^n, f_1^n, f_2^n, ...

3. Punctuation marks: (,), and the comma.

Most of our definitions and results will be formulated for the full standard language of first-order logic. However, depending on the application, we may also restrict the language to only a few predicate symbols, constant symbols, and function symbols.

Definition 2.1 (Language). A first-order *language* $\mathcal{L}$ is determined by its non-logical symbols: a set of predicate symbols, a set of constant symbols, and a set of function symbols, each function and predicate symbol having a fixed *arity* $n \geq 1$ (the number of arguments it takes). Formally:

$$\mathcal{L} = \langle \mathcal{C}, \mathcal{F}, \mathcal{R}, \mathrm{ar} \rangle$$

where $\mathcal{C}$ is a set of constant symbols, $\mathcal{F}$ is a set of function symbols, $\mathcal{R}$ is a set of predicate symbols, and ar assigns arities to the function and predicate symbols. The logical symbols (connectives, quantifiers, variables, identity, and punctuation) are shared across all languages.

Example 2.2. The language $\mathcal{L}_A$ of arithmetic contains a single two-place predicate symbol $<$, a single constant symbol o, one one-place function symbol $\prime$, and two two-place function symbols $+$ and $\times$. Officially: $\mathcal{L}_A = \langle \{\mathsf{o}\}, \{\prime, +, \times\}, \{<\}, \mathrm{ar} \rangle$ with $\mathrm{ar}(\prime) = 1$ and $\mathrm{ar}(+) = \mathrm{ar}(\times) = \mathrm{ar}(<) = 2$.

Remark 2. We treat all the propositional operators and both quantifiers as primitive symbols of the language. One could instead choose a smaller stock of primitive symbols and treat the other logical operators as defined. Truth-functionally complete sets of Boolean operators include $\{\neg, \lor\}$, $\{\neg, \land\}$, and $\{\neg, \rightarrow\}$; any one of these, combined with either quantifier, yields an expressively complete first-order language. In this text, we keep all connectives and both quantifiers as primitive for convenience.

2.2 Terms and Formulas

Once a first-order language $\mathcal{L}$ is given, we can define expressions built up from the basic vocabulary of $\mathcal{L}$. These include in particular *terms* and *formulas*.

Definition 2.3 (Terms). The set of *terms* $\mathrm{Trm}(\mathcal{L})$ of $\mathcal{L}$ is defined inductively by:

1. Every variable is a term.
2. Every constant symbol of $\mathcal{L}$ is a term.
3. If f is an n-place function symbol and $t_1, \dots, t_n$ are terms, then $f(t_1, \dots, t_n)$ is a term.
4. Nothing else is a term.

A term containing no variables is a *closed term*.

Definition 2.4 (Formulas)**.** The set of *formulas* $\mathrm{Frm}(\mathcal{L})$ of the language $\mathcal{L}$ is defined inductively as follows:

1. $\bot$ is an atomic formula.
2. $\top$ is an atomic formula.
3. If R is an n-place predicate symbol of $\mathcal{L}$ and t_1, …, t_n are terms of $\mathcal{L}$, then $R(t_1, \dots, t_n)$ is an atomic formula.
4. If t_1 and t_2 are terms of $\mathcal{L}$, then $=(t_1, t_2)$ is an atomic formula.
5. If φ is a formula, then $\lnot\varphi$ is a formula.
6. If φ and ψ are formulas, then $(\varphi \land \psi)$ is a formula.
7. If φ and ψ are formulas, then $(\varphi \lor \psi)$ is a formula.
8. If φ and ψ are formulas, then $(\varphi \rightarrow \psi)$ is a formula.
9. If φ and ψ are formulas, then $(\varphi \leftrightarrow \psi)$ is a formula.
10. If φ is a formula and x is a variable, then $\forall x\, \varphi$ is a formula.
11. If φ is a formula and x is a variable, then $\exists x\, \varphi$ is a formula.
12. Nothing else is a formula.

By convention, we write $=$ between its arguments and leave out the parentheses: $t_1 = t_2$ is an abbreviation for $=(t_1, t_2)$. Moreover, $\lnot{=}(t_1, t_2)$ is abbreviated as $t_1 \neq t_2$. When writing a formula $(\psi * \chi)$ constructed from ψ, χ using a two-place connective $*$, we will often leave out the outermost pair of parentheses and write simply $\psi * \chi$.

We write $\varphi \equiv \psi$ to express syntactic identity between strings of symbols, i.e., $\varphi \equiv \psi$ iff φ and ψ are strings of symbols of the same length and which contain the same symbol in each place.

Remark 3 (Propositional fragment). Propositional (PL) formulas are the fragment of FOL formulas built from propositional variables $p_0, p_1, \ldots$ using only connectives—no quantifiers, function symbols, or predicate symbols appear. Formally, the set $\mathrm{Frm}(\mathcal{L}_0)$ of PL formulas is defined by: (i) every propositional variable p_i is an atomic formula; (ii) the formation rules for connectives apply as above (clauses 5–9); (iii) the quantifier clauses are omitted.

As terms and formulas are built up from basic elements via inductive definitions, we can use the following induction principles to prove things about them.

Lemma 2.5 (Principle of induction on terms)**.** *Let $\mathcal{L}$ be a first-order language. If some property P is such that*

1. *it holds for every variable v,*
2. *it holds for every constant symbol a of $\mathcal{L}$, and*
3. *it holds for $f(t_1, \ldots, t_n)$ whenever it holds for $t_1, \ldots, t_n$ and f is an n-place function symbol of $\mathcal{L}$*

(assuming $t_1, \ldots, t_n$ are terms of $\mathcal{L}$), then P holds for every term in $\mathrm{Trm}(\mathcal{L})$.

Lemma 2.6 (Principle of induction on formulas)**.** *Let $\mathcal{L}$ be a first-order language. If some property P holds for all the atomic formulas and is such that*

1. *it holds for $\neg\varphi$ whenever it holds for φ;*
2. *it holds for $(\varphi \wedge \psi)$ whenever it holds for φ and ψ;*
3. *it holds for $(\varphi \vee \psi)$ whenever it holds for φ and ψ;*
4. *it holds for $(\varphi \rightarrow \psi)$ whenever it holds for φ and ψ;*
5. *it holds for $(\varphi \leftrightarrow \psi)$ whenever it holds for φ and ψ;*
6. *it holds for $\exists x\, \varphi$ whenever it holds for φ;*
7. *it holds for $\forall x\, \varphi$ whenever it holds for φ;*

(assuming φ and ψ are formulas of $\mathcal{L}$), then P holds for all formulas in $\mathrm{Frm}(\mathcal{L})$.

It is often useful to talk about the formulas that "make up" a given formula. We call these its *subformulas*. Any formula counts as a subformula of itself; a subformula of φ other than φ itself is a *proper subformula*.

Definition 2.7 (Immediate subformula)**.** If φ is a formula, the *immediate subformulas* of φ are defined as follows:

1. Atomic formulas have no immediate subformulas.
2. If $\varphi \equiv \neg\psi$, the only immediate subformula of φ is ψ.
3. If $\varphi \equiv (\psi * \chi)$, the immediate subformulas of φ are ψ and χ ($*$ is any one of the two-place connectives).
4. If $\varphi \equiv \forall x\, \psi$, the only immediate subformula of φ is ψ.
5. If $\varphi \equiv \exists x\, \psi$, the only immediate subformula of φ is ψ.

Definition 2.8 (Proper subformula)**.** If φ is a formula, the *proper subformulas* of φ are defined recursively as follows:

1. Atomic formulas have no proper subformulas.
2. If $\varphi \equiv \neg\psi$, the proper subformulas of φ are ψ together with all proper subformulas of ψ.
3. If $\varphi \equiv (\psi * \chi)$, the proper subformulas of φ are ψ, χ, together with all proper subformulas of ψ and those of χ.
4. If $\varphi \equiv \forall x\, \psi$, the proper subformulas of φ are ψ together with all proper subformulas of ψ.
5. If $\varphi \equiv \exists x\, \psi$, the proper subformulas of φ are ψ together with all proper subformulas of ψ.

Definition 2.9 (Subformula)**.** The subformulas of φ are φ itself together with all its proper subformulas.

Remark 4. The subformula relation is transitive: if χ is a subformula of ψ and ψ is a subformula of φ, then χ is a subformula of φ.

Definition 2.10 (Subterm)**.** The *subterms* of a term t are defined recursively:

1. t is a subterm of t.

2. If $f(t_1, \dots, t_n)$ is a subterm of s, then each t_i is a subterm of s.

Definition 2.11 (Main operator)**.** The *main operator* of a formula φ is defined as follows:

1. If φ is atomic, φ has no main operator.
2. If $\varphi \equiv \neg\psi$, the main operator of φ is $\neg$.
3. If $\varphi \equiv (\psi \wedge \chi)$, the main operator of φ is $\wedge$.
4. If $\varphi \equiv (\psi \vee \chi)$, the main operator of φ is $\vee$.
5. If $\varphi \equiv (\psi \rightarrow \chi)$, the main operator of φ is $\rightarrow$.
6. If $\varphi \equiv (\psi \leftrightarrow \chi)$, the main operator of φ is $\leftrightarrow$.
7. If $\varphi \equiv \forall x\, \psi$, the main operator of φ is $\forall$.
8. If $\varphi \equiv \exists x\, \psi$, the main operator of φ is $\exists$.

Definition 2.12 (Formula complexity)**.** The *complexity* (or *rank*) of a formula φ, written $\mathrm{cx}(\varphi)$, is a natural number defined by structural recursion:

1. If φ is atomic, then $\mathrm{cx}(\varphi) = 0$.
2. $\mathrm{cx}(\neg\psi) = \mathrm{cx}(\psi) + 1$.
3. $\mathrm{cx}((\psi * \chi)) = \max(\mathrm{cx}(\psi), \mathrm{cx}(\chi)) + 1$, where $*$ is any binary connective.
4. $\mathrm{cx}(\forall x\, \psi) = \mathrm{cx}(\psi) + 1$.
5. $\mathrm{cx}(\exists x\, \psi) = \mathrm{cx}(\psi) + 1$.

For instance, $\mathrm{cx}(P(x)) = 0$, $\mathrm{cx}(\neg P(x)) = 1$, and $\mathrm{cx}(P(x) \rightarrow \forall y\, Q(y)) = 2$.

An alternative, bottom-up approach to the construction of formulas uses *formation sequences*, which make explicit the step-by-step process by which a formula is built.

Definition 2.13 (Formation sequences for terms)**.** A finite sequence of $\mathcal{L}$-strings $\langle t_0, \dots, t_n \rangle$ is a *formation sequence* for a term t if $t \equiv t_n$ and for all $i \leq n$, either

t_i is a variable or a constant symbol, or $\mathcal{L}$ contains a k-ary function symbol f and there exist $m_0, \ldots, m_k < i$ such that $t_i \equiv f(t_{m_0}, \ldots, t_{m_k})$.

Definition 2.14 (Formation sequences for formulas)**.** A finite sequence of $\mathcal{L}$-strings $\langle \varphi_0, \ldots, \varphi_n \rangle$ is a *formation sequence* for φ if $\varphi \equiv \varphi_n$ and for all $i \leq n$, either φ_i is an atomic formula or there exist $j, k < i$ and a variable x such that one of the following holds:

1. $\varphi_i \equiv \neg\varphi_j$.
2. $\varphi_i \equiv (\varphi_j \land \varphi_k)$.
3. $\varphi_i \equiv (\varphi_j \lor \varphi_k)$.
4. $\varphi_i \equiv (\varphi_j \rightarrow \varphi_k)$.
5. $\varphi_i \equiv (\varphi_j \leftrightarrow \varphi_k)$.
6. $\varphi_i \equiv \forall x\, \varphi_j$.
7. $\varphi_i \equiv \exists x\, \varphi_j$.

Theorem 2.15. $\mathrm{Frm}(\mathcal{L})$ *is the set of all $\mathcal{L}$-strings φ such that there exists a formula formation sequence for φ.*

Proof sketch. One direction follows by induction on formulas: every formula in $\mathrm{Frm}(\mathcal{L})$ has a formation sequence (concatenate sequences for the immediate subformulas, then append the formula itself). The converse is proved by strong induction on the length of the formation sequence. □

2.3 Variables and Scope

Definition 2.16 (Free occurrences of a variable)**.** The *free* occurrences of a variable in a formula are defined inductively as follows:

1. If φ is atomic, all variable occurrences in φ are free.
2. If $\varphi \equiv \neg\psi$, the free variable occurrences of φ are exactly those of ψ.
3. If $\varphi \equiv (\psi * \chi)$, the free variable occurrences of φ are those in ψ together with those in χ.

4. If $\varphi \equiv \forall x\, \psi$, the free variable occurrences in φ are all of those in ψ except for occurrences of x.
5. If $\varphi \equiv \exists x\, \psi$, the free variable occurrences in φ are all of those in ψ except for occurrences of x.

Definition 2.17 (Free variables)**.** The set $\mathrm{FV}(()\varphi)$ of *free variables* of a formula φ is the set of variables that have at least one free occurrence in φ. Explicitly:

1. If $\varphi \equiv R(t_1, \dots, t_n)$, then $\mathrm{FV}(()\varphi) = \mathrm{Var}(t_1) \cup \cdots \cup \mathrm{Var}(t_n)$, where $\mathrm{Var}(t)$ is the set of variables occurring in term t.
2. $\mathrm{FV}(()\neg\varphi) = \mathrm{FV}(()\varphi)$.
3. $\mathrm{FV}(()(\varphi * \psi)) = \mathrm{FV}(()\varphi) \cup \mathrm{FV}(()\psi)$, where $*$ is any binary connective.
4. $\mathrm{FV}(()\forall x\, \varphi) = \mathrm{FV}(()\varphi) \setminus \{x\}$.
5. $\mathrm{FV}(()\exists x\, \varphi) = \mathrm{FV}(()\varphi) \setminus \{x\}$.

Definition 2.18 (Bound variables)**.** An occurrence of a variable in a formula φ is *bound* if it is not free.

Definition 2.19 (Scope)**.** If $\forall x\, \psi$ is an occurrence of a subformula in a formula φ, then the corresponding occurrence of ψ in φ is called the *scope* of the corresponding occurrence of $\forall x$. Similarly for $\exists x$.

If ψ is the scope of a quantifier occurrence $\forall x$ or $\exists x$ in φ, then the free occurrences of x in ψ are bound in $\forall x\, \psi$ and $\exists x\, \psi$, respectively. We say that these occurrences are *bound by* the mentioned quantifier occurrence.

Example 2.20. Consider the formula φ:

$$\forall v_0\, \underbrace{(A^1_0(v_0) \to A^2_0(v_0, v_1))}_{\psi} \to \exists v_1\, \underbrace{(A^2_1(v_0, v_1) \lor \forall v_0\, \overbrace{\neg A^1_1(v_0)}^{\theta})}_{\chi}$$

ψ is the scope of the first $\forall v_0$, χ is the scope of $\exists v_1$, and θ is the scope of the second $\forall v_0$. The first $\forall v_0$ binds the occurrences of v_0 in ψ, $\exists v_1$ binds the occurrence of v_1 in χ, and the second $\forall v_0$ binds the occurrence of v_0 in θ. The first occurrence of v_1 and the third occurrence of v_0 are free in φ. The last occurrence of v_0 is free in θ, but bound in χ and φ.

Definition 2.21 (Sentence)**.** A formula φ is a *sentence* iff it contains no free occurrences of variables; equivalently, iff $\mathrm{FV}(()\varphi) = \emptyset$.

2.4 Substitution

Definition 2.22 (Substitution in a term)**.** We define $s[t/x]$, the result of *substituting* t for every occurrence of x in s, recursively:

1. If $s \equiv c$ for a constant symbol c, then $s[t/x]$ is just s.
2. If $s \equiv y$ where y is a variable and $y \not\equiv x$, then $s[t/x]$ is also just s.
3. If $s \equiv x$, then $s[t/x]$ is t.
4. If $s \equiv f(t_1, \ldots, t_n)$, then $s[t/x]$ is $f(t_1[t/x], \ldots, t_n[t/x])$.

Definition 2.23 (Free for)**.** A term t is *free for* x in φ if none of the free occurrences of x in φ occur in the scope of a quantifier that binds a variable in t.

Example 2.24.

1. v_8 is free for v_1 in $\exists v_3 A^2_4(v_3, v_1)$.
2. $f^2_1(v_1, v_2)$ is *not* free for v_0 in $\forall v_2 A^2_4(v_0, v_2)$.

Definition 2.25 (Substitution in a formula)**.** If φ is a formula, x is a variable, and t is a term free for x in φ, then $\varphi[t/x]$ is the result of substituting t for all free occurrences of x in φ.

1. If $\varphi \equiv \bot$, then $\varphi[t/x]$ is $\bot$.
2. If $\varphi \equiv \top$, then $\varphi[t/x]$ is $\top$.
3. If $\varphi \equiv P(t_1, \ldots, t_n)$, then $\varphi[t/x]$ is $P(t_1[t/x], \ldots, t_n[t/x])$.
4. If $\varphi \equiv t_1 = t_2$, then $\varphi[t/x]$ is $t_1[t/x] = t_2[t/x]$.
5. If $\varphi \equiv \neg\psi$, then $\varphi[t/x]$ is $\neg\psi[t/x]$.
6. If $\varphi \equiv (\psi \wedge \chi)$, then $\varphi[t/x]$ is $(\psi[t/x] \wedge \chi[t/x])$.
7. If $\varphi \equiv (\psi \vee \chi)$, then $\varphi[t/x]$ is $(\psi[t/x] \vee \chi[t/x])$.

8. If $\varphi \equiv (\psi \to \chi)$, then $\varphi[t/x]$ is $(\psi[t/x] \to \chi[t/x])$.

9. If $\varphi \equiv (\psi \leftrightarrow \chi)$, then $\varphi[t/x]$ is $(\psi[t/x] \leftrightarrow \chi[t/x])$.

10. If $\varphi \equiv \forall y\, \psi$, then $\varphi[t/x]$ is $\forall y\, \psi[t/x]$, provided y is a variable other than x; otherwise $\varphi[t/x]$ is just φ.

11. If $\varphi \equiv \exists y\, \psi$, then $\varphi[t/x]$ is $\exists y\, \psi[t/x]$, provided y is a variable other than x; otherwise $\varphi[t/x]$ is just φ.

Note that substitution may be vacuous: if x does not occur in φ at all, then $\varphi[t/x]$ is just φ.

The restriction that t must be free for x in φ is necessary to exclude cases where a free variable in t is "captured" by a quantifier upon substitution. For instance, if $\varphi \equiv \exists y\, x < y$ and $t \equiv y$, then $\varphi[t/x]$ would be $\exists y\, y < y$—the free variable y has been captured by $\exists y$, which changes the meaning. We prevent this by requiring that none of the free variables in t end up bound by a quantifier in φ.

We use the following convention: if φ is a formula which may contain the variable x free, we write $\varphi(x)$; then $\varphi(t)$ is short for $\varphi[t/x]$.

Definition 2.26 (Simultaneous substitution)**.** Let φ be a formula, let $x_1, \dots, x_n$ be distinct variables, and let $t_1, \dots, t_n$ be terms. The *simultaneous substitution* $\varphi[t_1/x_1, \dots, t_n/x_n]$ is the result of replacing, in a single step, every free occurrence of each x_i in φ by the corresponding term t_i. The substitution is proper provided each t_i is free for x_i in φ.

Simultaneous substitution differs from iterated single substitution when the substituted terms contain variables that are themselves being replaced. For example, $P(x,y)[y/x, x/y] = P(y,x)$ (the variables are swapped), whereas the iterated substitution $P(x,y)[y/x][x/y] = P(y,y)[x/y] = P(x,x)$ gives a different result.

Definition 2.27 (Alphabetic variant)**.** Two formulas φ and ψ are *alphabetic variants* (written $\varphi \equiv_\alpha \psi$) if one can be obtained from the other by consistently renaming bound variables, without introducing variable capture. That is, φ and ψ differ only in the names of their bound variables, with each renaming being a consistent bijection on bound variable names that does not cause any free variable to become bound. For instance, $\forall x\, P(x)$ and $\forall y\, P(y)$ are alphabetic variants, since replacing the bound variable x by y throughout gives one formula from the other.

Remark 5 (Uniform substitution in PL). In propositional logic, *uniform substitution* replaces every occurrence of a propositional variable p_i by a formula ψ

throughout φ. Formally, $\varphi[\psi/p_i]$ denotes the result of replacing each occurrence of p_i by ψ in φ. This is a purely propositional operation: since propositional variables have no internal structure, there is no notion of "free for" and no risk of variable capture. In the FOL setting, uniform substitution is subsumed by the general substitution operation (see DEF-SYN001, §2.25).

The syntactic operations defined so far—substitution, formation, alphabetic variance—apply to arbitrary first-order languages. We now specialise to the *language of arithmetic* and classify its formulas by quantifier complexity.

2.5 Arithmetic Hierarchy

In the language of arithmetic $\mathcal{L}_A$, it is useful to single out syntactic classes of formulas defined by the pattern of quantifiers they contain. These classes—the Δ_0, Σ_1, and Π_1 formulas—play a central role in the study of incompleteness.

Definition 2.28 (Bounded quantification)**.** A *bounded existential formula* is one of the form $\exists x\,(x < t \land \varphi(x))$ where t is any term, which we conventionally write as $(\exists x < t)\ \varphi(x)$. A *bounded universal formula* is one of the form $\forall x\,(x < t \rightarrow \varphi(x))$ where t is any term, which we conventionally write as $(\forall x < t)\ \varphi(x)$.

Definition 2.29 (Δ_0, Σ_1, and Π_1 formulas)**.**

1. A formula ψ is Δ_0 if it is built up from atomic formulas using only propositional connectives and bounded quantification.

2. A formula φ is Σ_1 if $\varphi \equiv \exists x\, \psi(x)$ where ψ is Δ_0.

3. A formula φ is Π_1 if $\varphi \equiv \forall x\, \psi(x)$ where ψ is Δ_0.

2.6 Theorems

The way we defined formulas guarantees that every formula has a *unique reading*, i.e., there is essentially only one way of constructing it according to our formation rules for formulas and only one way of "interpreting" it. If this were not so, we would have ambiguous formulas, i.e., formulas that have more than one reading or interpretation—and that is clearly something we want to avoid. But more importantly, without this property, most of the definitions and proofs we are going to give will not go through.

Perhaps the best way to see why unique readability matters is to consider what would happen with bad formation rules. For instance, if we omitted parentheses and allowed: “If φ and ψ are formulas, then so is $\varphi \to \psi$,” then starting from an atomic formula θ we could form $\theta \to \theta \to \theta$ in two ways: taking $\varphi = \theta$ and $\psi = \theta \to \theta$, or taking $\varphi = \theta \to \theta$ and $\psi = \theta$. This would make the main operator ambiguous (the first vs. the second occurrence of $\to$), and recursive definitions on formulas would be ill-defined.

Lemma 2.30. *The number of left and right parentheses in a formula φ are equal.*

Proof. By induction on the construction of φ. Let $l(\varphi)$ be the number of left parentheses and $r(\varphi)$ the number of right parentheses in φ, and $l(t)$, $r(t)$ similarly for terms.

For atomic formulas: if $\varphi \equiv \bot$ or $\varphi \equiv \top$, then both counts are 0. If $\varphi \equiv R(t_1, \dots, t_n)$, then $l(\varphi) = 1 + l(t_1) + \cdots + l(t_n) = 1 + r(t_1) + \cdots + r(t_n) = r(\varphi)$, using the fact that $l(t) = r(t)$ for any term t. If $\varphi \equiv t_1 = t_2$, then $l(\varphi) = l(t_1) + l(t_2) = r(t_1) + r(t_2) = r(\varphi)$.

For the inductive cases: if $\varphi \equiv \neg\psi$, then by hypothesis $l(\psi) = r(\psi)$, so $l(\varphi) = l(\psi) = r(\psi) = r(\varphi)$. If $\varphi \equiv (\psi * \chi)$, then $l(\varphi) = 1 + l(\psi) + l(\chi) = 1 + r(\psi) + r(\chi) = r(\varphi)$. If $\varphi \equiv \forall x\, \psi$ or $\varphi \equiv \exists x\, \psi$, then $l(\varphi) = l(\psi) = r(\psi) = r(\varphi)$. □

Proposition 2.31 (Unique readability for atomic formulas). *If φ is an atomic formula, then it satisfies one, and only one of the following conditions.*

1. $\varphi \equiv \bot$.
2. $\varphi \equiv \top$.
3. $\varphi \equiv R(t_1, \dots, t_n)$ *where R is an n-place predicate symbol, $t_1, \dots, t_n$ are terms, and each of R, $t_1, \dots, t_n$ is uniquely determined.*
4. $\varphi \equiv t_1 = t_2$ *where t_1 and t_2 are uniquely determined terms.*

Proposition 2.32 (Unique Readability). *Every formula satisfies one, and only one of the following conditions.*

1. *φ is atomic.*
2. *φ is of the form $\neg\psi$.*
3. *φ is of the form $(\psi \wedge \chi)$.*
4. *φ is of the form $(\psi \vee \chi)$.*
5. *φ is of the form $(\psi \to \chi)$.*

6. *φ is of the form $(\psi \leftrightarrow \chi)$.*

7. *φ is of the form $\forall x\, \psi$.*

8. *φ is of the form $\exists x\, \psi$.*

Moreover, in each case ψ, or ψ and χ, are uniquely determined. This means that, e.g., there are no different pairs ψ, χ and ψ', χ' so that φ is both of the form $(\psi \to \chi)$ and $(\psi' \to \chi')$.

Proof sketch. The proof proceeds in four steps:

1. *Balanced parentheses*: Every formula has equal numbers of left and right parentheses (proved above).

2. *Prefix-freeness*: No proper prefix of a formula is itself a formula. (Proved by induction: every proper initial segment of a formula has strictly more left parentheses than right ones.)

3. *Unique atomic parsing*: Atomic formulas are uniquely parsed (Proposition 2.31).

4. *Main result*: Suppose $\varphi \equiv (\psi * \chi)$ and also $\varphi \equiv (\psi' *' \chi')$. If $\psi \equiv \psi'$, then clearly $* = *'$ and $\chi \equiv \chi'$. Otherwise, one of ψ, ψ' is a proper prefix of the other, contradicting prefix-freeness.

□

Remark 6 (PL unique readability). Unique readability holds equally for propositional formulas in $\mathrm{Frm}(\mathcal{L}_0)$, since PL formulas are a special case of FOL formulas (with the quantifier clauses vacuously absent).

Theorem 2.33 (Structural induction and recursion principles)**.**

1. ***Induction.*** *If a property P holds for all atomic formulas and is preserved by every formation rule (negation, binary connectives, quantification), then P holds for all formulas. Formally:*

$$[\forall \text{ atomic } \varphi\ P(\varphi)] \ \wedge\ [\forall \varphi\ (P(\varphi) \to P(\neg\varphi))] \ \wedge$$
$$[\forall \varphi \forall \psi\ (P(\varphi) \wedge P(\psi) \to P((\varphi * \psi)))] \ \wedge\ [\forall \varphi \forall x\ (P(\varphi) \to P(\forall x\, \varphi))]$$
$$\to\ \forall \varphi\ P(\varphi).$$

2. ***Recursion.*** *Any definition by structural recursion on formulas—specifying the value for atomic formulas and how to combine values across each formation rule—determines a* unique *function on* $\mathrm{Frm}(\mathcal{L})$.

Proof sketch. Part (1) follows from the fact that $\mathrm{Frm}(\mathcal{L})$ is inductively defined: it is the smallest set of strings closed under the formation rules. If the set $S = \{\varphi \in \mathrm{Frm}(\mathcal{L}) : P(\varphi)\}$ is also closed under the formation rules, then $\mathrm{Frm}(\mathcal{L}) \subseteq S$, so P holds for all formulas.

Part (2) follows from unique readability (Proposition 2.32): since each formula has a unique outermost construction step, the recursive clauses never conflict, and the function is well-defined. $\square$

Chapter 3

Semantics

3.1 Structures

First-order languages are, by themselves, *uninterpreted*: the constant symbols, function symbols, and predicate symbols have no specific meaning attached to them. Meanings are given by specifying a *structure*. It specifies the *domain*, i.e., the objects which the constant symbols pick out, the function symbols operate on, and the quantifiers range over. In addition, it specifies which constant symbols pick out which objects, how a function symbol maps objects to objects, and which objects the predicate symbols apply to. Structures are the basis for *semantic* notions in logic, e.g., the notion of consequence, validity, satisfiability. They are variously called "structures," "interpretations," or "models" in the literature.

Definition 3.1 (Structures)**.** A *structure* $\mathfrak{M}$ for a language $\mathcal{L}$ of first-order logic consists of the following elements:

1. *Domain:* a non-empty set, $|\mathfrak{M}|$.
2. *Interpretation of constant symbols:* for each constant symbol c of $\mathcal{L}$, an element $c^{\mathfrak{M}} \in |\mathfrak{M}|$.
3. *Interpretation of predicate symbols:* for each n-place predicate symbol R of $\mathcal{L}$ (other than $=$), an n-place relation $R^{\mathfrak{M}} \subseteq |\mathfrak{M}|^n$.
4. *Interpretation of function symbols:* for each n-place function symbol f of $\mathcal{L}$, an n-place function $f^{\mathfrak{M}}\colon |\mathfrak{M}|^n \to |\mathfrak{M}|$.

Example 3.2. A structure $\mathfrak{M}$ for the language of arithmetic consists of a set, an element of $|\mathfrak{M}|$, $\mathsf{o}^{\mathfrak{M}}$, as interpretation of the constant symbol o, a one-place function $\prime^{\mathfrak{M}}\colon |\mathfrak{M}| \to |\mathfrak{M}|$, two two-place functions $+^{\mathfrak{M}}$ and $\times^{\mathfrak{M}}$, both $|\mathfrak{M}|^2 \to |\mathfrak{M}|$, and a two-place relation $<^{\mathfrak{M}} \subseteq |\mathfrak{M}|^2$.

An obvious example of such a structure is the following:

1. $|\mathfrak{N}| = \mathbb{N}$
2. $o^{\mathfrak{N}} = 0$
3. $\prime^{\mathfrak{N}}(n) = n + 1$ for all $n \in \mathbb{N}$
4. $+^{\mathfrak{N}}(n, m) = n + m$ for all $n, m \in \mathbb{N}$
5. $\times^{\mathfrak{N}}(n, m) = n \cdot m$ for all $n, m \in \mathbb{N}$
6. $<^{\mathfrak{N}} = \{\langle n, m \rangle : n \in \mathbb{N}, m \in \mathbb{N}, n < m\}$

The structure $\mathfrak{N}$ for $\mathcal{L}_A$ so defined is called the *standard model of arithmetic*, because it interprets the non-logical constants of $\mathcal{L}_A$ exactly how you would expect.

The stipulations we make as to what counts as a structure impact our logic. For example, the choice to prevent empty domains ensures that $\exists x\,(\varphi(x) \lor \neg\varphi(x))$ is valid. Allowing empty domains or names that do not refer leads to *free logic*.[1]

We can assign values to closed terms (terms containing no variables) using only the structure, without needing a variable assignment.

Definition 3.3 (Value of closed terms)**.** If t is a closed term of the language $\mathcal{L}$ and $\mathfrak{M}$ is a structure for $\mathcal{L}$, the *value* $\mathrm{Val}^{\mathfrak{M}}(t)$ is defined as follows:

1. If t is just the constant symbol c, then $\mathrm{Val}^{\mathfrak{M}}(c) = c^{\mathfrak{M}}$.
2. If t is of the form $f(t_1, \ldots, t_n)$, then
$$\mathrm{Val}^{\mathfrak{M}}(t) = f^{\mathfrak{M}}(\mathrm{Val}^{\mathfrak{M}}(t_1), \ldots, \mathrm{Val}^{\mathfrak{M}}(t_n)).$$

A structure is *covered* if every element of the domain is the value of some closed term.

3.2 Satisfaction and Truth

The basic notion that relates expressions such as terms and formulas, on the one hand, and structures on the other, are those of *value* of a term and *satisfaction* of a formula. Informally, the value of a term is an element of a structure—if the term is just a constant, its value is the object assigned to the constant

[1]In free logic, existential generalization requires an additional premise: $\varphi(a)$ and $\exists x\, x = a$, therefore $\exists x\, \varphi(x)$.

by the structure, and if it is built up using function symbols, the value is computed from the values of constants and the functions assigned to the functions in the term. A formula is *satisfied* in a structure if the interpretation given to the predicates makes the formula true in the domain of the structure. This notion of satisfaction is specified inductively: the specification of the structure directly states when atomic formulas are satisfied, and we define when a complex formula is satisfied depending on the main connective or quantifier and whether or not the immediate subformulas are satisfied.

The case of the quantifiers is a bit tricky, as the immediate subformula of a quantified formula has a free variable, and structures don't specify the values of variables. In order to deal with this difficulty, we introduce *variable assignments* and define satisfaction not with respect to a structure alone, but with respect to a structure plus a variable assignment.

Definition 3.4 (Variable Assignment)**.** A *variable assignment s* for a structure $\mathfrak{M}$ is a function which maps each variable to an element of $|\mathfrak{M}|$, i.e., $s\colon \mathrm{Var} \to |\mathfrak{M}|$.

A structure assigns a value to each constant symbol, and a variable assignment to each variable. But we want to use terms built up from them to also name elements of the domain. For this we define the value of terms inductively. For constant symbols and variables the value is just as the structure or the variable assignment specifies it; for more complex terms it is computed recursively using the functions the structure assigns to the function symbols.

Definition 3.5 (Value of Terms)**.** If t is a term of the language $\mathcal{L}$, $\mathfrak{M}$ is a structure for $\mathcal{L}$, and s is a variable assignment for $\mathfrak{M}$, the *value* $\mathrm{Val}_s^{\mathfrak{M}}(t)$ is defined as follows:

1. If t is a constant symbol c, then $\mathrm{Val}_s^{\mathfrak{M}}(c) = c^{\mathfrak{M}}$.
2. If t is a variable x, then $\mathrm{Val}_s^{\mathfrak{M}}(x) = s(x)$.
3. If $t = f(t_1, \dots, t_n)$, then

$$\mathrm{Val}_s^{\mathfrak{M}}(t) = f^{\mathfrak{M}}(\mathrm{Val}_s^{\mathfrak{M}}(t_1), \dots, \mathrm{Val}_s^{\mathfrak{M}}(t_n)).$$

Definition 3.6 (x-Variant)**.** If s is a variable assignment for a structure $\mathfrak{M}$, then any variable assignment s' for $\mathfrak{M}$ which differs from s at most in what it assigns to x is called an *x-variant* of s. If s' is an x-variant of s we write $s' \sim_x s$.

Note that an x-variant of an assignment s does not *have* to assign something different to x. In fact, every assignment counts as an x-variant of itself.

Definition 3.7. If s is a variable assignment for a structure $\mathfrak{M}$ and $m \in |\mathfrak{M}|$, then the assignment $s[m/x]$ is the variable assignment defined by

$$s[m/x](y) = \begin{cases} m & \text{if } y \equiv x \\ s(y) & \text{otherwise.} \end{cases}$$

In other words, $s[m/x]$ is the particular x-variant of s which assigns the domain element m to x, and assigns the same things to variables other than x that s does.

Definition 3.8 (Satisfaction)**.** Satisfaction of a formula φ in a structure $\mathfrak{M}$ relative to a variable assignment s, in symbols: $\mathfrak{M}, s \vDash \varphi$, is defined recursively as follows. (We write $\mathfrak{M}, s \nvDash \varphi$ to mean "not $\mathfrak{M}, s \vDash \varphi$.")

1. $\mathfrak{M}, s \vDash R(t_1, \dots, t_n)$ iff $\langle \mathrm{Val}_s^{\mathfrak{M}}(t_1), \dots, \mathrm{Val}_s^{\mathfrak{M}}(t_n) \rangle \in R^{\mathfrak{M}}$.
2. $\mathfrak{M}, s \vDash t_1 = t_2$ iff $\mathrm{Val}_s^{\mathfrak{M}}(t_1) = \mathrm{Val}_s^{\mathfrak{M}}(t_2)$.
3. $\mathfrak{M}, s \vDash \neg\psi$ iff $\mathfrak{M}, s \nvDash \psi$.
4. $\mathfrak{M}, s \vDash (\psi \wedge \chi)$ iff $\mathfrak{M}, s \vDash \psi$ and $\mathfrak{M}, s \vDash \chi$.
5. $\mathfrak{M}, s \vDash (\psi \vee \chi)$ iff $\mathfrak{M}, s \vDash \psi$ or $\mathfrak{M}, s \vDash \chi$ (or both).
6. $\mathfrak{M}, s \vDash (\psi \rightarrow \chi)$ iff $\mathfrak{M}, s \nvDash \psi$ or $\mathfrak{M}, s \vDash \chi$ (or both).
7. $\mathfrak{M}, s \vDash (\psi \leftrightarrow \chi)$ iff either both $\mathfrak{M}, s \vDash \psi$ and $\mathfrak{M}, s \vDash \chi$, or neither $\mathfrak{M}, s \vDash \psi$ nor $\mathfrak{M}, s \vDash \chi$.
8. $\mathfrak{M}, s \vDash \forall x\, \psi$ iff for every element $m \in |\mathfrak{M}|$, $\mathfrak{M}, s[m/x] \vDash \psi$.
9. $\mathfrak{M}, s \vDash \exists x\, \psi$ iff for at least one element $m \in |\mathfrak{M}|$, $\mathfrak{M}, s[m/x] \vDash \psi$.

The variable assignments are important in the quantifier clauses. We cannot define satisfaction of $\forall x\, \psi(x)$ by "for all $m \in |\mathfrak{M}|$, $\mathfrak{M} \vDash \psi(m)$," because if $m \in |\mathfrak{M}|$, it is not a symbol of the language, and so $\psi(m)$ is not a formula (that is, $\psi[m/x]$ is undefined). We also cannot assume that we have constant symbols or terms available that name every element of $\mathfrak{M}$, since there is nothing in the definition of structures that requires it. Variable assignments allow us to link variables directly with elements of the domain, resolving this difficulty.

Remark 7 (PL Specialization). For propositional logic, a *truth-value assignment* (or *valuation*) $v\colon \mathrm{PropVar} \to \{0, 1\}$ replaces the structure-plus-variable-assignment pair. Satisfaction reduces to: $v \vDash p$ iff $v(p) = 1$; the connective clauses are the same as above (without quantifier clauses). Propositional satisfaction is thus the quantifier-free fragment of first-order satisfaction.

The value of a term t, and whether or not a formula φ is satisfied in a structure with respect to s, only depend on the assignments s makes to the variables in t and the free variables of φ.

The value of a term depends only on the values of its variables: if s_1 and s_2 agree on all variables occurring in t, then $\mathrm{Val}^{\mathfrak{M}}_{s_1}(t) = \mathrm{Val}^{\mathfrak{M}}_{s_2}(t)$.

Proposition 3.9. *If the free variables in φ are among $x_1, \dots, x_n$, and $s_1(x_i) = s_2(x_i)$ for $i = 1, \dots, n$, then $\mathfrak{M}, s_1 \vDash \varphi$ iff $\mathfrak{M}, s_2 \vDash \varphi$.*

Proof sketch. By induction on the complexity of φ. The base case uses the corresponding result for terms. For connectives, apply the induction hypothesis to subformulas (whose free variables are among those of φ). For the quantifier case $\exists x\, \psi$: if $\mathfrak{M}, s_1[m/x] \vDash \psi$ for some m, then $s_1[m/x]$ and $s_2[m/x]$ agree on the free variables of ψ (which are among $x_1, \dots, x_n, x$), so the induction hypothesis gives $\mathfrak{M}, s_2[m/x] \vDash \psi$. Similarly for $\forall x\, \psi$. □

Sentences have no free variables, so any two variable assignments assign the same things to all the (zero) free variables of any sentence. Therefore:

Corollary 3.10. *If φ is a sentence and s a variable assignment, then $\mathfrak{M}, s \vDash \varphi$ iff $\mathfrak{M}, s' \vDash \varphi$ for every variable assignment s'.*

This justifies the following definition.

Definition 3.11 (Truth in a Structure)**.** If φ is a sentence, we say that a structure $\mathfrak{M}$ *satisfies* φ, $\mathfrak{M} \vDash \varphi$, iff $\mathfrak{M}, s \vDash \varphi$ for all variable assignments s.

If $\mathfrak{M} \vDash \varphi$, we also simply say that φ *is true in* $\mathfrak{M}$.

Definition 3.12 (Satisfaction for sets of sentences)**.** If Γ is a set of sentences, we say that a structure $\mathfrak{M}$ *satisfies* Γ, $\mathfrak{M} \vDash \Gamma$, iff $\mathfrak{M} \vDash \varphi$ for all $\varphi \in \Gamma$.

Proposition 3.13. *Let $\mathfrak{M}$ be a structure, φ be a sentence, and s a variable assignment. $\mathfrak{M} \vDash \varphi$ iff $\mathfrak{M}, s \vDash \varphi$.*

3.3 Validity and Consequence

Given the definition of structures for first-order languages, we can define some basic semantic properties of and relationships between sentences. The simplest of these is the notion of *validity* of a sentence. A sentence is valid if it is satisfied in every structure. Valid sentences are those that are satisfied regardless of how the non-logical symbols in it are interpreted. Valid sentences are therefore also called *logical truths*—they are true, i.e., satisfied, in any structure and hence their truth depends only on the logical symbols occurring in them and their syntactic structure, but not on the non-logical symbols or their interpretation.

Definition 3.14 (Validity)**.** A sentence φ is *valid*, $\vDash \varphi$, iff $\mathfrak{M} \vDash \varphi$ for every structure $\mathfrak{M}$.

Definition 3.15 (Entailment)**.** A set of sentences Γ *entails* a sentence φ, $\Gamma \vDash \varphi$, iff for every structure $\mathfrak{M}$ with $\mathfrak{M} \vDash \Gamma$, $\mathfrak{M} \vDash \varphi$.

Definition 3.16 (Satisfiability)**.** A set of sentences Γ is *satisfiable* if $\mathfrak{M} \vDash \Gamma$ for some structure $\mathfrak{M}$. If Γ is not satisfiable it is called *unsatisfiable*.

Definition 3.17 (Model)**.** Let Γ be a set of sentences in a language $\mathcal{L}$. We say that a structure $\mathfrak{M}$ *is a model of* Γ if $\mathfrak{M} \vDash \varphi$ for all $\varphi \in \Gamma$.

Definition 3.18 (Semantic Consistency)**.** A set of sentences Γ is *semantically consistent* if it is satisfiable, i.e., if it has at least one model. This is the semantic counterpart of syntactic consistency (see §4.1).

We record some basic relationships:

- $\Gamma \vDash \varphi$ iff $\Gamma \cup \{\neg\varphi\}$ is unsatisfiable.
- (Monotonicity) If $\Gamma \subseteq \Gamma'$ and $\Gamma \vDash \varphi$, then $\Gamma' \vDash \varphi$.
- (Semantic Deduction Theorem) $\Gamma \cup \{\varphi\} \vDash \psi$ iff $\Gamma \vDash \varphi \rightarrow \psi$.

Remark 8 (PL Specialization: Tautology). A propositional formula is a *tautology* if it is true under every truth-value assignment. This specializes first-order validity to propositional structures: a tautology is a formula valid under all valuations $v\colon \mathrm{PropVar} \to \{0,1\}$.

Remark 9 (PL Specialization: PL Consequence). PL entailment $\Gamma \vDash_{\mathrm{PL}} \varphi$ specializes first-order entailment to propositional structures: every truth-value assignment making all formulas in Γ true also makes φ true.

3.4 Models and Theories

Definition 3.19 (Semantic Completeness of a Theory)**.** A theory T is *semantically complete* iff for every sentence φ in its language, either $T \vDash \varphi$ or $T \vDash \neg\varphi$.

Note that semantic completeness of a theory is *not* the same as the Completeness Theorem (see §6.2), which states that $\Gamma \vDash \varphi$ implies $\Gamma \vdash \varphi$.

Definition 3.20 (Theory of a Structure)**.** Given a structure $\mathfrak{M}$, the *theory* of $\mathfrak{M}$ is the set $\mathrm{Th}(\mathfrak{M})$ of sentences that are true in $\mathfrak{M}$, i.e., $\mathrm{Th}(\mathfrak{M}) = \{\varphi : \mathfrak{M} \vDash \varphi\}$.

We also use the term “theory” informally to refer to sets of sentences having an intended interpretation, whether deductively closed or not.

Proposition 3.21. *For any* $\mathfrak{M}$, $\mathrm{Th}(\mathfrak{M})$ *is semantically complete.*

Proof. For any sentence φ either $\mathfrak{M} \vDash \varphi$ or $\mathfrak{M} \vDash \neg\varphi$, so either $\varphi \in \mathrm{Th}(\mathfrak{M})$ or $\neg\varphi \in \mathrm{Th}(\mathfrak{M})$. □

Definition 3.22 (Definable Set)**.** A subset of $|\mathfrak{M}|^n$ is *definable* (in $\mathfrak{M}$) if there is a formula $\varphi(x_1, \dots, x_n)$ with free variables $x_1, \dots, x_n$ such that the subset equals $\{\langle a_1, \dots, a_n\rangle \in |\mathfrak{M}|^n : \mathfrak{M}, s[a_1/x_1], \dots, s[a_n/x_n] \vDash \varphi\}$.

Definition 3.23 (Elementary Equivalence)**.** Given two structures $\mathfrak{M}$ and $\mathfrak{M}'$ for the same language $\mathcal{L}$, we say that $\mathfrak{M}$ is *elementarily equivalent to* $\mathfrak{M}'$, written $\mathfrak{M} \equiv \mathfrak{M}'$, if and only if for every sentence φ of $\mathcal{L}$, $\mathfrak{M} \vDash \varphi$ iff $\mathfrak{M}' \vDash \varphi$.

Proposition 3.24. *If $\mathfrak{N} \vDash \varphi$ for every $\varphi \in \text{Th}(\mathfrak{M})$, then $\mathfrak{M} \equiv \mathfrak{N}$.*

Proof. Since $\mathfrak{N} \vDash \varphi$ for all $\varphi \in \text{Th}(\mathfrak{M})$, $\text{Th}(\mathfrak{M}) \subseteq \text{Th}(\mathfrak{N})$. If $\mathfrak{N} \vDash \varphi$, then $\mathfrak{N} \nvDash \neg\varphi$, so $\neg\varphi \notin \text{Th}(\mathfrak{M})$. Since $\text{Th}(\mathfrak{M})$ is complete, $\varphi \in \text{Th}(\mathfrak{M})$. So, $\text{Th}(\mathfrak{N}) \subseteq \text{Th}(\mathfrak{M})$, and we have $\mathfrak{M} \equiv \mathfrak{N}$. □

Remark 10. Consider $\mathfrak{R} = \langle \mathbb{R}, < \rangle$, the structure whose domain is the set $\mathbb{R}$ of the real numbers, in the language comprising only a 2-place predicate symbol interpreted as the $<$ relation over the reals. Clearly $\mathfrak{R}$ is uncountable; however, since $\text{Th}(\mathfrak{R})$ is obviously consistent, by the Löwenheim–Skolem theorem (see §6.4) it has a countable model, say $\mathfrak{S}$, and by the above proposition, $\mathfrak{R} \equiv \mathfrak{S}$. Moreover, since $\mathfrak{R}$ and $\mathfrak{S}$ are not isomorphic, this shows that the converse of the Isomorphism Lemma (Theorem 3.33) fails in general: elementary equivalence does not imply isomorphism.

Definition 3.25 (Axiomatized Theory)**.** A set of sentences Γ is *closed* iff, whenever $\Gamma \vDash \varphi$ then $\varphi \in \Gamma$. The *closure* of a set of sentences Γ is $\{\varphi : \Gamma \vDash \varphi\}$. We say that Γ is *axiomatized by* a set of sentences Δ if Γ is the closure of Δ.

3.5 Arithmetic Models

The structures studied in this section are models of the first-order theories of arithmetic (**Q**, **PA**) introduced in §4.6.

Definition 3.26 (Standard Model of Arithmetic)**.** The *standard model of arithmetic* is the structure $\mathfrak{N}$ defined as follows:

1. $|\mathfrak{N}| = \mathbb{N}$
2. $0^{\mathfrak{N}} = 0$
3. $\prime^{\mathfrak{N}}(n) = n + 1$ for all $n \in \mathbb{N}$
4. $+^{\mathfrak{N}}(n, m) = n + m$ for all $n, m \in \mathbb{N}$
5. $\times^{\mathfrak{N}}(n, m) = n \cdot m$ for all $n, m \in \mathbb{N}$
6. $<^{\mathfrak{N}} = \{\langle n, m \rangle : n \in \mathbb{N}, m \in \mathbb{N}, n < m\}$

Definition 3.27 (True Arithmetic)**.** The theory of *true arithmetic* is the set of sentences satisfied in the standard model of arithmetic, i.e.,

$$\mathbf{TA} = \{\varphi : \mathfrak{N} \vDash \varphi\}.$$

TA is a theory (closed under entailment), for whenever $\mathbf{TA} \vDash \varphi$, φ is satisfied in every structure which satisfies **TA**. Since $\mathfrak{N} \vDash \mathbf{TA}$, we have that $\mathfrak{N} \vDash \varphi$, and so $\varphi \in \mathbf{TA}$.

Remark 11 (Interpretability). Informally, an interpretation of a language $\mathcal{L}_1$ in another language $\mathcal{L}_2$ involves defining the universe, relation symbols, and function symbols of $\mathcal{L}_1$ with formulas in $\mathcal{L}_2$. One can show: if a theory **T** is consistent with the interpretation of Robinson's **Q** (see §4.6), then **T** is undecidable, and no consistent extension of **T** is decidable. In particular, there is no decidable or complete consistent axiomatizable extension of **ZFC**.

3.6 Model-Theoretic Concepts

Reducts and Expansions

Often it is useful or necessary to compare languages which have symbols in common, as well as structures for these languages. The most common case is when all the symbols in a language $\mathcal{L}$ are also part of a language $\mathcal{L}'$, i.e., $\mathcal{L} \subseteq \mathcal{L}'$.

Definition 3.28 (Reduct and Expansion)**.** Suppose $\mathcal{L} \subseteq \mathcal{L}'$, $\mathfrak{M}$ is an $\mathcal{L}$-structure and $\mathfrak{M}'$ is an $\mathcal{L}'$-structure. $\mathfrak{M}$ is the *reduct* of $\mathfrak{M}'$ to $\mathcal{L}$, and $\mathfrak{M}'$ is an *expansion* of $\mathfrak{M}$ to $\mathcal{L}'$ iff

1. $|\mathfrak{M}| = |\mathfrak{M}'|$;
2. For every constant symbol $c \in \mathcal{L}$, $c^{\mathfrak{M}} = c^{\mathfrak{M}'}$;
3. For every function symbol $f \in \mathcal{L}$, $f^{\mathfrak{M}} = f^{\mathfrak{M}'}$;
4. For every predicate symbol $P \in \mathcal{L}$, $P^{\mathfrak{M}} = P^{\mathfrak{M}'}$.

If $\mathfrak{M}$ is a reduct of $\mathfrak{M}'$, then for all $\mathcal{L}$-sentences φ, $\mathfrak{M} \vDash \varphi$ iff $\mathfrak{M}' \vDash \varphi$.

When we have an $\mathcal{L}$-structure $\mathfrak{M}$, and $\mathcal{L}' = \mathcal{L} \cup \{P\}$ is the expansion of $\mathcal{L}$ obtained by adding a single n-place predicate symbol P, and $R \subseteq |\mathfrak{M}|^n$ is an n-place relation, then we write $(\mathfrak{M}, R)$ for the expansion $\mathfrak{M}'$ of $\mathfrak{M}$ with $P^{\mathfrak{M}'} = R$.

Substructures

The domain of a structure $\mathfrak{M}$ may be a subset of another $\mathfrak{M}'$. But we should obviously only consider $\mathfrak{M}$ a "part" of $\mathfrak{M}'$ if not only $|\mathfrak{M}| \subseteq |\mathfrak{M}'|$, but $\mathfrak{M}$ and

$\mathfrak{M}'$ "agree" in how they interpret the symbols of the language at least on the shared part $|\mathfrak{M}|$.

Definition 3.29 (Substructure)**.** Given structures $\mathfrak{M}$ and $\mathfrak{M}'$ for the same language $\mathcal{L}$, we say that $\mathfrak{M}$ is a *substructure* of $\mathfrak{M}'$, and $\mathfrak{M}'$ an *extension* of $\mathfrak{M}$, written $\mathfrak{M} \subseteq \mathfrak{M}'$, iff

1. $|\mathfrak{M}| \subseteq |\mathfrak{M}'|$;
2. For each constant $c \in \mathcal{L}$, $c^{\mathfrak{M}} = c^{\mathfrak{M}'}$;
3. For each n-place function symbol $f \in \mathcal{L}$, $f^{\mathfrak{M}}(a_1, \dots, a_n) = f^{\mathfrak{M}'}(a_1, \dots, a_n)$ for all $a_1, \dots, a_n \in |\mathfrak{M}|$;
4. For each n-place predicate symbol $R \in \mathcal{L}$, $\langle a_1, \dots, a_n \rangle \in R^{\mathfrak{M}}$ iff $\langle a_1, \dots, a_n \rangle \in R^{\mathfrak{M}'}$ for all $a_1, \dots, a_n \in |\mathfrak{M}|$.

Remark 12. If the language contains no constant or function symbols, then any non-empty $N \subseteq |\mathfrak{M}|$ determines a substructure $\mathfrak{N}$ of $\mathfrak{M}$ with domain $|\mathfrak{N}| = N$ by putting $R^{\mathfrak{N}} = R^{\mathfrak{M}} \cap N^n$.

Homomorphisms

Definition 3.30 (Homomorphism)**.** A function $h\colon |\mathfrak{M}| \to |\mathfrak{M}'|$ between structures $\mathfrak{M}$ and $\mathfrak{M}'$ for the same language $\mathcal{L}$ is a *homomorphism* if:

1. For every constant symbol c: $h(c^{\mathfrak{M}}) = c^{\mathfrak{M}'}$;
2. For every n-place function symbol f: $h(f^{\mathfrak{M}}(a_1, \dots, a_n)) = f^{\mathfrak{M}'}(h(a_1), \dots, h(a_n))$;
3. For every n-place predicate symbol R: $\langle a_1, \dots, a_n \rangle \in R^{\mathfrak{M}} \Rightarrow \langle h(a_1), \dots, h(a_n) \rangle \in R^{\mathfrak{M}'}$.

A homomorphism is weaker than an isomorphism: it need not be bijective, and it preserves relations in one direction only (it need not reflect them).

Definition 3.31 (Embedding)**.** An *embedding* $h\colon |\mathfrak{M}| \hookrightarrow |\mathfrak{M}'|$ between structures $\mathfrak{M}$ and $\mathfrak{M}'$ for the same language $\mathcal{L}$ is an injective homomorphism that also *reflects* relations: for every n-place predicate symbol R,

$$\langle a_1, \dots, a_n \rangle \in R^{\mathfrak{M}} \quad \text{iff} \quad \langle h(a_1), \dots, h(a_n) \rangle \in R^{\mathfrak{M}'}.$$

Every isomorphism is a surjective embedding; every embedding is a homomorphism that additionally reflects atomic formulas.

Isomorphism

First-order structures can be alike in one of two ways. One way in which they can be alike is that they make the same sentences true—we call such structures *elementarily equivalent* (Definition 3.23). But structures can be very different and still make the same sentences true—for instance, one can be countable and the other not. So another, stricter, aspect in which structures can be alike is if they are fundamentally the same, in the sense that they only differ in the objects that make them up, but not in their structural features.

Definition 3.32 (Isomorphism)**.** Given two structures $\mathfrak{M}$ and $\mathfrak{M}'$ for the same language $\mathcal{L}$, we say that $\mathfrak{M}$ is *isomorphic to* $\mathfrak{M}'$, written $\mathfrak{M} \simeq \mathfrak{M}'$, if and only if there is a function $h \colon |\mathfrak{M}| \to |\mathfrak{M}'|$ such that:

1. h is injective: if $h(x) = h(y)$ then $x = y$;
2. h is surjective: for every $y \in |\mathfrak{M}'|$ there is $x \in |\mathfrak{M}|$ such that $h(x) = y$;
3. For every constant symbol c: $h(c^{\mathfrak{M}}) = c^{\mathfrak{M}'}$;
4. For every n-place predicate symbol P:
$$\langle a_1, \dots, a_n \rangle \in P^{\mathfrak{M}} \quad \text{iff} \quad \langle h(a_1), \dots, h(a_n) \rangle \in P^{\mathfrak{M}'};$$
5. For every n-place function symbol f:
$$h(f^{\mathfrak{M}}(a_1, \dots, a_n)) = f^{\mathfrak{M}'}(h(a_1), \dots, h(a_n)).$$

Theorem 3.33 (Isomorphism Lemma)**.** *If $\mathfrak{M} \simeq \mathfrak{M}'$ then $\mathfrak{M} \equiv \mathfrak{M}'$.*

Proof. Let h be an isomorphism of $\mathfrak{M}$ onto $\mathfrak{M}'$. For any assignment s, $h \circ s$ is the composition of h and s, i.e., the assignment in $\mathfrak{M}'$ such that $(h \circ s)(x) = h(s(x))$. By induction on t and φ one proves the stronger claims:

a. $h(\mathrm{Val}_s^{\mathfrak{M}}(t)) = \mathrm{Val}_{h \circ s}^{\mathfrak{M}'}(t)$.

b. $\mathfrak{M}, s \vDash \varphi$ iff $\mathfrak{M}', h \circ s \vDash \varphi$.

Part (a) is proved by induction on the complexity of t.

1. If $t \equiv c$, then $h(\mathrm{Val}_s^{\mathfrak{M}}(c)) = h(c^{\mathfrak{M}}) = c^{\mathfrak{M}'} = \mathrm{Val}_{h \circ s}^{\mathfrak{M}'}(c)$.

2. If $t \equiv x$, then $h(\mathrm{Val}_s^{\mathfrak{M}}(x)) = h(s(x)) = (h \circ s)(x) = \mathrm{Val}_{h\circ s}^{\mathfrak{M}'}(x)$.
3. If $t \equiv f(t_1, \dots, t_n)$, then by induction hypothesis $h(\mathrm{Val}_s^{\mathfrak{M}}(t_i)) = \mathrm{Val}_{h\circ s}^{\mathfrak{M}'}(t_i)$ for each i, so

$$\begin{aligned} h(\mathrm{Val}_s^{\mathfrak{M}}(t)) &= h(f^{\mathfrak{M}}(\mathrm{Val}_s^{\mathfrak{M}}(t_1), \dots, \mathrm{Val}_s^{\mathfrak{M}}(t_n))) \\ &= f^{\mathfrak{M}'}(h(\mathrm{Val}_s^{\mathfrak{M}}(t_1)), \dots, h(\mathrm{Val}_s^{\mathfrak{M}}(t_n))) \\ &= f^{\mathfrak{M}'}(\mathrm{Val}_{h\circ s}^{\mathfrak{M}'}(t_1), \dots, \mathrm{Val}_{h\circ s}^{\mathfrak{M}'}(t_n)) \\ &= \mathrm{Val}_{h\circ s}^{\mathfrak{M}'}(t). \end{aligned}$$

Part (b) is proved by induction on the complexity of φ. If φ is a sentence, the assignments s and $h \circ s$ are irrelevant, and we have $\mathfrak{M} \vDash \varphi$ iff $\mathfrak{M}' \vDash \varphi$. □

Definition 3.34. An *automorphism* of a structure $\mathfrak{M}$ is an isomorphism of $\mathfrak{M}$ onto itself.

Definition 3.35 (Elementary Substructure)**.** $\mathfrak{M}$ is an *elementary substructure* of $\mathfrak{M}'$, written $\mathfrak{M} \preccurlyeq \mathfrak{M}'$, if $\mathfrak{M}$ is a substructure of $\mathfrak{M}'$ and for every formula $\varphi(x_1, \dots, x_n)$ and all $a_1, \dots, a_n \in |\mathfrak{M}|$: $\mathfrak{M}, a_1, \dots, a_n \vDash \varphi$ iff $\mathfrak{M}', a_1, \dots, a_n \vDash \varphi$.

Diagrams

Definition 3.36 (Diagram)**.** Let $\mathfrak{M}$ be a structure for $\mathcal{L}$. Expand $\mathcal{L}$ to $\mathcal{L}_M = \mathcal{L} \cup \{c_a : a \in |\mathfrak{M}|\}$ by adding a new constant symbol c_a for each element a of the domain. The *atomic diagram* of $\mathfrak{M}$ is the set

$$\mathrm{Diag}(\mathfrak{M}) = \{\varphi : \varphi \text{ is atomic or negated atomic in } \mathcal{L}_M \text{ and } \mathfrak{M} \vDash \varphi\}.$$

The *elementary diagram* of $\mathfrak{M}$ is the set

$$\mathrm{ElDiag}(\mathfrak{M}) = \{\varphi : \varphi \in \mathrm{Sent}(\mathcal{L}_M) \text{ and } \mathfrak{M} \vDash \varphi\}.$$

Any model of $\mathrm{Diag}(\mathfrak{M})$ contains an isomorphic copy of $\mathfrak{M}$ (via the map $a \mapsto c_a^{\mathfrak{M}}$), and any model of $\mathrm{ElDiag}(\mathfrak{M})$ contains an elementary extension of $\mathfrak{M}$.

Definition 3.37 (Complete Type)**.** Let $\mathfrak{M}$ be a structure for $\mathcal{L}$ and $A \subseteq |\mathfrak{M}|$. A *complete n-type over A* is a maximal consistent set p of formulas $\varphi(x_1, \dots, x_n)$ with parameters from A that is finitely satisfiable in $\mathfrak{M}$. The set of all complete

n-types over A is denoted $S_n^{\mathfrak{M}}(A)$. Types classify the possible "behaviors" of n-tuples in models extending $\mathfrak{M}$.

Ultraproducts

Definition 3.38 (Ultraproduct)**.** Given a family of structures $\{\mathfrak{M}_i\}_{i\in I}$ for the same language $\mathcal{L}$ and an ultrafilter $\mathcal{U}$ on I, the *ultraproduct* $\prod_{i\in I} \mathfrak{M}_i/\mathcal{U}$ is the structure with domain

$$\prod_{i\in I} |\mathfrak{M}_i| / {\sim_{\mathcal{U}}},$$

where $f \sim_{\mathcal{U}} g$ iff $\{i \in I : f(i) = g(i)\} \in \mathcal{U}$. Constant, function, and predicate symbols are interpreted componentwise modulo $\mathcal{U}$. When all $\mathfrak{M}_i = \mathfrak{M}$, the construction is called an *ultrapower* of $\mathfrak{M}$.

Ultraproducts provide a purely model-theoretic proof of compactness (see CP-003, §6.3): a set of sentences is satisfiable iff every finite subset is, by taking an ultraproduct of the finite-subset models. This avoids completeness entirely.

Categoricity and Dense Linear Orders

Definition 3.39 (Categoricity)**.** A theory T is *κ-categorical* if all models of T of cardinality κ are isomorphic. By the Löwenheim–Skolem theorem, no theory with infinite models is categorical in all cardinalities.

Definition 3.40 (Dense Linear Ordering)**.** A *dense linear ordering without endpoints* is a structure $\mathfrak{M}$ for the language containing a single 2-place predicate symbol $<$ satisfying the following sentences:

1. $\forall x\, \neg x < x$ (irreflexivity)
2. $\forall x\, \forall y\, \forall z\, (x < y \rightarrow (y < z \rightarrow x < z))$ (transitivity)
3. $\forall x\, \forall y\, (x < y \lor x = y \lor y < x)$ (totality)
4. $\forall x\, \exists y\, x < y$ (no greatest element)
5. $\forall x\, \exists y\, y < x$ (no least element)
6. $\forall x\, \forall y\, (x < y \rightarrow \exists z\, (x < z \land z < y))$ (density)

Theorem 3.41 (Cantor)**.** *Any two countable dense linear orderings without endpoints are isomorphic.*

Proof. Let $\mathfrak{M}_1$ and $\mathfrak{M}_2$ be countable dense linear orderings without endpoints, with $<_1 = <^{\mathfrak{M}_1}$ and $<_2 = <^{\mathfrak{M}_2}$, and let $\mathcal{I}$ be the set of all partial isomorphisms between them. $\mathcal{I}$ is not empty since at least $\emptyset \in \mathcal{I}$. We show that $\mathcal{I}$ satisfies the Back-and-Forth property.

To show $\mathcal{I}$ satisfies the Forth property, let $p \in \mathcal{I}$ and let $p(a_i) = b_i$ for $i = 1$, $\dots$, n, and without loss of generality suppose $a_1 <_1 a_2 <_1 \dots <_1 a_n$. Given $a \in |\mathfrak{M}_1|$, find $b \in |\mathfrak{M}_2|$ as follows:

1. if $a <_1 a_1$ let $b \in |\mathfrak{M}_2|$ be such that $b <_2 b_1$;
2. if $a_n <_1 a$ let $b \in |\mathfrak{M}_2|$ be such that $b_n <_2 b$;
3. if $a_i <_1 a <_1 a_{i+1}$ for some i, then let $b \in |\mathfrak{M}_2|$ be such that $b_i <_2 b <_2 b_{i+1}$.

It is always possible to find a b with the desired property since $\mathfrak{M}_2$ is a dense linear ordering without endpoints. Define $q = p \cup \{\langle a, b\rangle\}$ so that $q \in \mathcal{I}$ is the desired extension of p. The Back property is similar. By the back-and-forth theorem (applied to countable structures), $\mathfrak{M}_1 \simeq \mathfrak{M}_2$. □

The theory of dense linear orders without endpoints is thus $\aleph_0$-categorical.

Remark 13. Let $\mathfrak{S}$ be any countable dense linear ordering without endpoints. Then by Cantor's theorem, $\mathfrak{S} \simeq \mathfrak{Q}$, where $\mathfrak{Q} = (\mathbb{Q}, <)$. Now consider the structure $\mathfrak{R} = (\mathbb{R}, <)$ from Remark 10. There is a countable structure $\mathfrak{S}$ such that $\mathfrak{R} \equiv \mathfrak{S}$. But $\mathfrak{S}$ is a countable dense linear ordering without endpoints, and so it is isomorphic (and hence elementarily equivalent) to $\mathfrak{Q}$. By transitivity of elementary equivalence, $\mathfrak{R} \equiv \mathfrak{Q}$.

Standard and Non-Standard Models of Arithmetic

A structure for $\mathcal{L}_A$ is *standard* if it is isomorphic to $\mathfrak{N}$.

Proposition 3.42. *If a structure $\mathfrak{M}$ is standard, then its domain is the set of values of the standard numerals, i.e., $|\mathfrak{M}| = \{\mathrm{Val}^{\mathfrak{M}}(\overline{n}) : n \in \mathbb{N}\}$.*

Proof sketch. Since $\mathfrak{M}$ is standard, there is an isomorphism $g \colon \mathbb{N} \to |\mathfrak{M}|$. Then $g(n) = g(\mathrm{Val}^{\mathfrak{N}}(\overline{n})) = \mathrm{Val}^{\mathfrak{M}}(\overline{n})$, and g is surjective. □

Proposition 3.43. *If $\mathfrak{M} \vDash \mathbf{Q}$, and $|\mathfrak{M}| = \{\mathrm{Val}^{\mathfrak{M}}(\overline{n}) : n \in \mathbb{N}\}$, then $\mathfrak{M}$ is standard.*

Proof sketch. The function $g(n) = \mathrm{Val}^{\mathfrak{M}}(\overline{n})$ is surjective by hypothesis and injective because $\mathbf{Q} \vdash \overline{n} \neq \overline{m}$ whenever $n \neq m$, so $\mathfrak{M} \vDash \overline{n} \neq \overline{m}$. One verifies that g is an isomorphism by checking that $\mathbf{Q}$ proves the relevant identities for 0, $\prime$, $+$, $\times$, and $<$. □

A structure $\mathfrak{M}$ for $\mathcal{L}_A$ is *non-standard* if it is not isomorphic to $\mathfrak{N}$. The elements $x \in |\mathfrak{M}|$ which are equal to $\mathrm{Val}^{\mathfrak{M}}(\overline{n})$ for some $n \in \mathbb{N}$ are called *standard numbers* (of $\mathfrak{M}$), and those not, *non-standard numbers*.

If a structure $\mathfrak{M}$ for $\mathcal{L}_A$ contains a non-standard number, $\mathfrak{M}$ is non-standard.

Proposition 3.44. **TA** *has a countable non-standard model.*

Proof. Expand $\mathcal{L}_A$ by a new constant symbol c and consider the set of sentences

$$\Gamma = \mathbf{TA} \cup \{c \neq \overline{0}, c \neq \overline{1}, c \neq \overline{2}, \dots\}$$

Any model $\mathfrak{M}^c$ of Γ would contain an element $x = c^{\mathfrak{M}}$ which is non-standard, since $x \neq \mathrm{Val}^{\mathfrak{M}}(\overline{n})$ for all $n \in \mathbb{N}$. Also, obviously, $\mathfrak{M}^c \vDash \mathbf{TA}$, since $\mathbf{TA} \subseteq \Gamma$. If we turn $\mathfrak{M}^c$ into a structure $\mathfrak{M}$ for $\mathcal{L}_A$ simply by forgetting about c, its domain still contains the non-standard x, and also $\mathfrak{M} \vDash \mathbf{TA}$.

We use the compactness theorem to show that Γ has a model. Consider any finite subset $\Gamma_0 \subseteq \Gamma$. Suppose k is the largest number so that $c \neq \overline{k} \in \Gamma_0$. Define $\mathfrak{N}_k$ by expanding $\mathfrak{N}$ to include the interpretation $c^{\mathfrak{N}_k} = k+1$. Then $\mathfrak{N}_k \vDash \Gamma_0$, since c does not occur in **TA**, and $\mathrm{Val}^{\mathfrak{N}_k}(c) = k+1 \neq n$ for $n \leq k$. Thus every finite subset of Γ is satisfiable, so by compactness, Γ is satisfiable. □

In a non-standard model $\mathfrak{M}$ of **PA**, the ordering $<^{\mathfrak{M}}$ is a linear strict order. The element $0^{\mathfrak{M}}$ is least, every element has a unique successor and (except for $0^{\mathfrak{M}}$) a unique predecessor, and all standard elements are less than all non-standard elements.

Every non-standard element x is contained in a *block* $[x]$ consisting of all elements reachable from x by finitely many applications of successor and predecessor. Each block has no least and no greatest element. Distinct blocks are disjoint and respect the ordering: if $x < y$ and $[x] \neq [y]$, then $u < v$ for all $u \in [x]$ and $v \in [y]$.

Proposition 3.45. *The ordering of the non-standard blocks is dense: if $x < y$ and $[x] \neq [y]$, then there is a block $[z]$ distinct from both that is between them.*

Proof sketch. **PA** proves that every element is divisible by 2 (possibly with remainder). If x is non-standard, the "average" z of x and y satisfies $x < z < y$ and $[z] \neq [x]$, $[z] \neq [y]$. □

The non-standard blocks are therefore ordered like the rationals: they form a countable dense linear ordering without endpoints. By Cantor's theorem (Theorem 3.41), any two such orderings are isomorphic. It follows that for any two countable non-standard models $\mathfrak{M}_1$ and $\mathfrak{M}_2$ of true arithmetic, their reducts to the language containing $<$ and $=$ only are isomorphic. However, they need not be isomorphic in the full language of arithmetic, as there are non-isomorphic ways to define addition and multiplication.

Definition 3.46 (Computable Structure)**.** A structure $\mathfrak{M}$ for $\mathcal{L}_A$ is *computable* iff $|\mathfrak{M}| = \mathbb{N}$ and $\prime^{\mathfrak{M}}$, $+^{\mathfrak{M}}$, $\times^{\mathfrak{M}}$ are computable functions and $<^{\mathfrak{M}}$ is a decidable relation.

Theorem 3.47 (Tennenbaum's Theorem)**.** *$\mathfrak{N}$ is the only computable model of* **PA**.

Overspill

A classical application of the compactness theorem is the overspill principle.

Theorem 3.48 (Overspill)**.** *If a set Γ of sentences has arbitrarily large finite models, then it has an infinite model.*

Proof. Expand the language of Γ by adding countably many new constants c_0, c_1, … and consider $\Gamma \cup \{c_i \neq c_j : i \neq j\}$. Since Γ has arbitrarily large finite models, every finite subset of this expanded set is satisfiable. By compactness, the whole set has a model $\mathfrak{M}$ whose domain must be infinite, since it satisfies all inequalities $c_i \neq c_j$. □

Proposition 3.49. *There is no sentence φ of any first-order language that is true in a structure $\mathfrak{M}$ if and only if the domain $|\mathfrak{M}|$ of the structure is infinite.*

Proof. If there were such a φ, its negation $\neg\varphi$ would be true in all and only the finite structures, and it would therefore have arbitrarily large finite models but lack an infinite model, contradicting the overspill theorem. □

3.7 Theorems

The Coincidence Lemma

Extensionality, sometimes called relevance, can be expressed informally as follows: the only factors that bear upon the satisfaction of formula φ in a structure $\mathfrak{M}$ relative to a variable assignment s, are the size of the domain and the assignments made by $\mathfrak{M}$ and s to the elements of the language that actually appear in φ.

One immediate consequence is that where two structures $\mathfrak{M}$ and $\mathfrak{M}'$ agree on all the elements of the language appearing in a sentence φ and have the same domain, $\mathfrak{M}$ and $\mathfrak{M}'$ must also agree on whether or not φ itself is true.

Proposition 3.50 (Extensionality / Coincidence Lemma)**.** *Let φ be a formula, and $\mathfrak{M}_1$ and $\mathfrak{M}_2$ be structures with $|\mathfrak{M}_1| = |\mathfrak{M}_2|$, and s a variable assignment on $|\mathfrak{M}_1| = |\mathfrak{M}_2|$. If $c^{\mathfrak{M}_1} = c^{\mathfrak{M}_2}$, $R^{\mathfrak{M}_1} = R^{\mathfrak{M}_2}$, and $f^{\mathfrak{M}_1} = f^{\mathfrak{M}_2}$ for every constant symbol c, relation symbol R, and function symbol f occurring in φ, then $\mathfrak{M}_1, s \vDash \varphi$ iff $\mathfrak{M}_2, s \vDash \varphi$.*

Proof sketch. First prove (by induction on t) that for every term, $\mathrm{Val}_s^{\mathfrak{M}_1}(t) = \mathrm{Val}_s^{\mathfrak{M}_2}(t)$. Then prove the proposition by induction on φ, making use of the claim just proved for the induction basis (where φ is atomic). □

Corollary 3.51 (Extensionality for Sentences)**.** *Let φ be a sentence and $\mathfrak{M}_1$, $\mathfrak{M}_2$ as in the Coincidence Lemma. Then $\mathfrak{M}_1 \vDash \varphi$ iff $\mathfrak{M}_2 \vDash \varphi$.*

Proof. Follows from the Coincidence Lemma by Corollary 3.10. □

The Substitution Lemma

Moreover, the value of a term, and whether or not a structure satisfies a formula, only depend on the values of its subterms.

Proposition 3.52 (Substitution Lemma for Terms)**.** *Let $\mathfrak{M}$ be a structure, t and t' terms, and s a variable assignment. Then $\mathrm{Val}_s^{\mathfrak{M}}(t[t'/x]) = \mathrm{Val}_{s[\mathrm{Val}_s^{\mathfrak{M}}(t')/x]}^{\mathfrak{M}}(t)$.*

Proof. By induction on t.

1. If t is a constant, say, $t \equiv c$, then $t[t'/x] = c$, and $\mathrm{Val}_s^{\mathfrak{M}}(c) = c^{\mathfrak{M}} = \mathrm{Val}_{s[\mathrm{Val}_s^{\mathfrak{M}}(t')/x]}^{\mathfrak{M}}(c)$.

2. If t is a variable other than x, say, $t \equiv y$, then $t[t'/x] = y$, and $\mathrm{Val}_s^{\mathfrak{M}}(y) = \mathrm{Val}_{s[\mathrm{Val}_s^{\mathfrak{M}}(t')/x]}^{\mathfrak{M}}(y)$ since $s \sim_x s[\mathrm{Val}_s^{\mathfrak{M}}(t')/x]$.

3. If $t \equiv x$, then $t[t'/x] = t'$. But $\mathrm{Val}_{s[\mathrm{Val}_s^{\mathfrak{M}}(t')/x]}^{\mathfrak{M}}(x) = \mathrm{Val}_s^{\mathfrak{M}}(t')$ by definition of $s[\mathrm{Val}_s^{\mathfrak{M}}(t')/x]$.

4. If $t \equiv f(t_1, \dots, t_n)$ then by the definition of substitution and the induction hypothesis:

$$\begin{aligned} \mathrm{Val}_s^{\mathfrak{M}}(t[t'/x]) &= f^{\mathfrak{M}}(\mathrm{Val}_s^{\mathfrak{M}}(t_1[t'/x]), \dots, \mathrm{Val}_s^{\mathfrak{M}}(t_n[t'/x])) \\ &= f^{\mathfrak{M}}(\mathrm{Val}^{\mathfrak{M}}_{s[\mathrm{Val}_s^{\mathfrak{M}}(t')/x]}(t_1), \dots, \mathrm{Val}^{\mathfrak{M}}_{s[\mathrm{Val}_s^{\mathfrak{M}}(t')/x]}(t_n)) \\ &= \mathrm{Val}^{\mathfrak{M}}_{s[\mathrm{Val}_s^{\mathfrak{M}}(t')/x]}(t). \end{aligned}$$

□

Proposition 3.53 (Substitution Lemma for Formulas)**.** *Let $\mathfrak{M}$ be a structure, φ a formula, t' a term, and s a variable assignment. Then $\mathfrak{M}, s \vDash \varphi[t'/x]$ iff $\mathfrak{M}, s[\mathrm{Val}_s^{\mathfrak{M}}(t')/x] \vDash \varphi$.*

Proof. By induction on φ, parallel to the proof for terms. □

The point of the Substitution Lemma is the following. Suppose we have a term t or a formula φ and some term t', and we want to know the value of $t[t'/x]$ or whether or not $\varphi[t'/x]$ is satisfied in a structure $\mathfrak{M}$ relative to a variable assignment s. Then we can either perform the substitution first and then consider the value or satisfaction relative to $\mathfrak{M}$ and s, or we can first determine the value $m = \mathrm{Val}_s^{\mathfrak{M}}(t')$ of t' in $\mathfrak{M}$ relative to s, change the variable assignment to $s[m/x]$ and then consider the value of t in $\mathfrak{M}$ and $s[m/x]$, or whether $\mathfrak{M}, s[m/x] \vDash \varphi$. The Substitution Lemma guarantees that the answer will be the same, whichever way we do it.

Chapter 4

Deduction

4.1 Generic Proof Theory

This section develops proof-theoretic concepts at the *generic* level, independent of any particular proof system. Each concept defined here—axiom schema, rule of inference, derivation, provability, consistency, and so on—is instantiated differently by the concrete proof systems presented in DED.2–DED.5. The structural properties established below (reflexivity, monotonicity, transitivity, compactness) hold in each system, though the proofs differ in detail. Canonical proofs are given here for the Hilbert-style (axiomatic) case; remarks indicate how they adapt to other systems.

4.1.1 Axiom Schemas and Rules of Inference

Definition 4.1 (Axiom Schema)**.** An *axiom schema* is a schematic expression involving metavariables that stands for a (typically infinite) collection of formulas: every formula obtained by uniformly replacing the metavariables with concrete formulas is an *instance* of the schema and counts as a logical axiom.

For instance, the schema $\varphi \to (\psi \to \varphi)$ generates infinitely many axioms—one for each choice of formulas φ and ψ. Different proof systems employ different collections of axiom schemas (or none at all, as in the case of natural deduction).

Definition 4.2 (Non-Logical Axiom)**.** A *non-logical axiom* is a sentence adopted as a premise of a theory but not derivable from the logical axioms alone. A set of non-logical axioms is sometimes called a set of *proper axioms* or simply the *axioms of the theory*. A theory is *axiomatized* by a set Γ_0 of non-logical axioms when the theory consists of all sentences that follow (semantically or proof-theoretically) from Γ_0.

Definition 4.3 (Rule of Inference)**.** A *rule of inference* gives a sufficient condition for what counts as a correct inference step in a derivation from a set of assumptions Γ. More precisely, a rule specifies one or more *premises* and a *conclusion*; a step is correct when each premise either appears earlier in the derivation, is an axiom, or is an element of Γ.

The simplest and most ubiquitous rule of inference is *modus ponens*:

> If $\psi \to \varphi$ and ψ both occur (as axioms, assumptions, or earlier conclusions) in a derivation, then φ is a correct inference step.

What counts as a correct derivation depends on which rules of inference are admitted and on what is taken as an axiom. Different proof systems make different choices:

- *Axiomatic (Hilbert-style) deduction* (see DED.2) uses many axiom schemas and few rules (typically just modus ponens and a quantifier rule).
- *Natural deduction* (see DED.3) uses no logical axioms, but has introduction and elimination rules for each connective and quantifier.
- *Sequent calculus* (see DED.4) operates on sequents and employs left/right introduction rules together with structural rules.
- *Tableaux* (see DED.5) work by attempted refutation, applying branch-extension rules to signed formulas.

4.1.2 Proof Systems and Derivations

Definition 4.4 (Proof System)**.** A *proof system* is a specification of:

1. A set of logical axioms (possibly empty, possibly given by schemas);
2. A set of rules of inference;
3. A definition of what constitutes a *derivation*.

A proof system determines a derivability relation $\Gamma \vdash \varphi$ on formulas.

Definition 4.5 (Derivation)**.** Let Γ be a set of formulas of $\mathcal{L}$. A *derivation* from Γ is a finite combinatorial object—a sequence of formulas, a tree of formulas, or a tree of sequents, depending on the proof system—in which every step is justified either as:

1. an element of Γ (an assumption); or
2. an instance of a logical axiom; or
3. the conclusion of a correct application of a rule of inference to earlier steps.

A derivation *derives* its final formula (its last element, its root, or the succedent of its root sequent, depending on the system).

Remark 14. The shape of derivations differs across proof systems. In axiomatic deduction, a derivation is a finite *sequence* of formulas. In natural deduction and the sequent calculus, a derivation is a finite *tree*. In tableaux, a derivation is a finitely branching tree of signed formulas. Despite these differences, the abstract notion of derivability is uniform: $\Gamma \vdash \varphi$ means that there exists a derivation of φ from Γ in the given system.

4.1.3 Sequents and Structural Rules

Definition 4.6 (Sequent)**.** A *sequent* is an expression of the form

$$\Gamma \Rightarrow \Delta$$

where Γ and Δ are finite (possibly empty) sequences of sentences of the language $\mathcal{L}$. Γ is called the *antecedent* and Δ is called the *succedent*.

The intuitive reading of a sequent $\Gamma \Rightarrow \Delta$ is: if all of the sentences in the antecedent hold, then at least one of the sentences in the succedent holds. That is, if $\Gamma = \langle \varphi_1, \ldots, \varphi_m \rangle$ and $\Delta = \langle \psi_1, \ldots, \psi_n \rangle$, then $\Gamma \Rightarrow \Delta$ holds iff

$$(\varphi_1 \wedge \cdots \wedge \varphi_m) \rightarrow (\psi_1 \vee \cdots \vee \psi_n)$$

holds. When Γ is empty, $\Rightarrow \Delta$ asserts that $\psi_1 \vee \cdots \vee \psi_n$ holds. When Δ is empty, $\Gamma \Rightarrow$ asserts that $\neg(\varphi_1 \wedge \cdots \wedge \varphi_m)$.

Definition 4.7 (Structural Rules)**.** The *structural rules* of a proof system govern the manipulation of the context (the set or sequence of assumptions or side formulas) without introducing or eliminating any logical connective. The principal structural rules are:

1. **Weakening** (W): one may add a formula to the context without affecting derivability.

$$\frac{\Gamma \Rightarrow \Delta}{\varphi, \Gamma \Rightarrow \Delta}\ \text{WL} \qquad \frac{\Gamma \Rightarrow \Delta}{\Gamma \Rightarrow \Delta, \varphi}\ \text{WR}$$

2. **Contraction** (C): two copies of the same formula may be collapsed into one.

$$\frac{\varphi, \varphi, \Gamma \Rightarrow \Delta}{\varphi, \Gamma \Rightarrow \Delta}\ \text{CL} \qquad \frac{\Gamma \Rightarrow \Delta, \varphi, \varphi}{\Gamma \Rightarrow \Delta, \varphi}\ \text{CR}$$

3. **Exchange** (X): the order of formulas in the context may be permuted.

$$\frac{\Gamma, \varphi, \psi, \Pi \Rightarrow \Delta}{\Gamma, \psi, \varphi, \Pi \Rightarrow \Delta}\ \text{XL} \qquad \frac{\Gamma \Rightarrow \Delta, \varphi, \psi, \Lambda}{\Gamma \Rightarrow \Delta, \psi, \varphi, \Lambda}\ \text{XR}$$

4. **Cut** (Cut): if a formula can be derived on the right and consumed on the left, it may be eliminated.

$$\frac{\Gamma \Rightarrow \Delta, \varphi \qquad \varphi, \Pi \Rightarrow \Lambda}{\Gamma, \Pi \Rightarrow \Delta, \Lambda}\ \text{Cut}$$

Remark 15. The structural rules are stated here in their sequent-calculus form, but analogous phenomena occur in every proof system. In axiomatic deduction, weakening corresponds to monotonicity of derivability (adding unused hypotheses), and cut corresponds to transitivity (chaining derivations). In natural deduction, weakening is built into the assumption mechanism, contraction is implicit in re-use of assumptions, and cut corresponds to substituting a derivation for an assumption. Substructural logics arise by restricting or removing structural rules: e.g., linear logic drops weakening and contraction; relevant logic restricts weakening.

4.1.4 Assumption Discharge

Definition 4.8 (Assumption Discharge)**.** In natural deduction, an *assumption* is any sentence occupying a topmost (leaf) position in a derivation tree. Certain rules of inference—notably $\rightarrow$Intro, $\neg$Intro, $\vee$Elim, and $\exists$Elim—permit one to *discharge* assumptions: the discharged assumption is cancelled and no longer counts among the open (undischarged) assumptions of the derivation. The label notation $[\varphi]^n$ indicates that the assumption φ bearing label n is discharged by the corresponding inference step.

Discharging is a permission, not a requirement: one may apply a discharging rule even when the assumption to be discharged does not actually occur in the derivation above. The set of *undischarged* assumptions of a derivation is the set of assumptions that have not been cancelled by any rule application. A derivation with no undischarged assumptions is a *proof* (of a theorem).

4.1.5 Provability and Theorems

Definition 4.9 (Provability)**.** A formula φ is *derivable* from Γ, written $\Gamma \vdash \varphi$, if there is a derivation from Γ ending in φ (in whatever proof system is in force).

Definition 4.10 (Theorem)**.** A formula φ is a *theorem* if there is a derivation of φ from the empty set. We write $\vdash \varphi$ if φ is a theorem and $\nvdash \varphi$ if it is not.

4.1.6 Structural Properties of Derivability

Just as we have defined semantic notions (validity, entailment, satisfiability), we now establish corresponding *proof-theoretic properties*. These are not defined by appeal to satisfaction in structures, but by appeal to the derivability or non-derivability of formulas. It was an important discovery, the content of the *soundness* and *completeness* theorems (see CP-001, Soundness, §6.1 and CP-002, Completeness, §6.2), that these notions coincide.

Proposition 4.11 (Reflexivity)**.** *If $\varphi \in \Gamma$, then $\Gamma \vdash \varphi$.*

Proof. The formula φ by itself constitutes a (trivial) derivation of φ from Γ: in axiomatic deduction, it is a one-element sequence whose sole entry is justified as an element of Γ; in natural deduction, it is a single-node tree (an assumption); in the sequent calculus, the initial sequent $\varphi \Rightarrow \varphi$ suffices. □

Proposition 4.12 (Monotonicity)**.** *If $\Gamma \subseteq \Delta$ and $\Gamma \vdash \varphi$, then $\Delta \vdash \varphi$.*

Proof. Any derivation of φ from Γ is also a derivation of φ from Δ, since every element of Γ used in the derivation is also an element of Δ. □

Proposition 4.13 (Transitivity). *If $\Gamma \vdash \varphi$ and $\{\varphi\} \cup \Delta \vdash \psi$, then $\Gamma \cup \Delta \vdash \psi$.*

Proof. Suppose $\{\varphi\} \cup \Delta \vdash \psi$. Then there is a derivation $\psi_1, \dots, \psi_l = \psi$ from $\{\varphi\} \cup \Delta$. Some of the steps in that derivation will be correct because of a rule which refers to a prior line $\psi_i = \varphi$. By hypothesis, there is a derivation of φ from Γ, i.e., a derivation $\varphi_1, \dots, \varphi_k = \varphi$ where every φ_i is an axiom, an element of Γ, or correct by a rule of inference. Now consider the sequence

$$\varphi_1, \dots, \varphi_k = \varphi, \psi_1, \dots, \psi_l = \psi.$$

This is a correct derivation of ψ from $\Gamma \cup \Delta$ since every $\psi_i = \varphi$ is now justified by the same rule which justifies $\varphi_k = \varphi$. □

Remark 16. The proof above is stated for axiomatic deduction, where derivations are sequences and transitivity amounts to concatenation. In natural deduction, transitivity is realized by substituting the derivation of φ for the assumption φ in the derivation of ψ. In the sequent calculus, transitivity is an instance of the cut rule. Each proof system instantiates these concepts differently; see DED.2–DED.5.

Note that transitivity implies in particular: if $\Gamma \vdash \varphi$ and $\varphi \vdash \psi$, then $\Gamma \vdash \psi$. It follows also that if $\varphi_1, \dots, \varphi_n \vdash \psi$ and $\Gamma \vdash \varphi_i$ for each i, then $\Gamma \vdash \psi$.

Proposition 4.14 (Inconsistency Characterization). *Γ is inconsistent iff $\Gamma \vdash \varphi$ for every φ.*

Proof. If Γ is inconsistent, then $\Gamma \vdash \bot$. From $\bot$ any formula φ follows (by the logical axiom or rule $\bot \to \varphi$ in all standard proof systems). Conversely, if $\Gamma \vdash \varphi$ for every φ, then in particular $\Gamma \vdash \bot$, so Γ is inconsistent. □

Proposition 4.15 (Compactness).

1. *If $\Gamma \vdash \varphi$ then there is a finite subset $\Gamma_0 \subseteq \Gamma$ such that $\Gamma_0 \vdash \varphi$.*
2. *If every finite subset of Γ is consistent, then Γ is consistent.*

Proof.

1. If $\Gamma \vdash \varphi$, then there is a derivation—a finite object. Let Γ_0 be the set of elements of Γ that actually appear in the derivation. Since the derivation is finite, Γ_0 is finite, and the derivation is equally a derivation from Γ_0. So $\Gamma_0 \vdash \varphi$.
2. This is the contrapositive of (1) for the special case $\varphi \equiv \bot$.

□

4.1.7 Consistency and Derivability

We now establish a number of properties of the derivability relation. They are independently interesting, and each plays a role in the proof of the completeness theorem (see CP-002, Completeness, §6.2).

Definition 4.16 (Syntactic Consistency)**.** A set Γ of formulas is *consistent* if and only if $\Gamma \nvdash \bot$; it is *inconsistent* otherwise.

Proposition 4.17. *If $\Gamma \vdash \varphi$ and $\Gamma \cup \{\varphi\}$ is inconsistent, then Γ is inconsistent.*

Proof. If $\Gamma \cup \{\varphi\}$ is inconsistent, then $\Gamma \cup \{\varphi\} \vdash \bot$. By reflexivity, $\Gamma \vdash \psi$ for every $\psi \in \Gamma$. Since also $\Gamma \vdash \varphi$ by hypothesis, $\Gamma \vdash \psi$ for every $\psi \in \Gamma \cup \{\varphi\}$. By transitivity, $\Gamma \vdash \bot$, i.e., Γ is inconsistent. □

Proposition 4.18. *$\Gamma \vdash \varphi$ iff $\Gamma \cup \{\neg\varphi\}$ is inconsistent.*

Proof. First suppose $\Gamma \vdash \varphi$. Then $\Gamma \cup \{\neg\varphi\} \vdash \varphi$ by monotonicity, and $\Gamma \cup \{\neg\varphi\} \vdash \neg\varphi$ by reflexivity. Since from φ and $\neg\varphi$ together we can derive $\bot$ (via the axiom or rule $\neg\varphi \to (\varphi \to \bot)$ and modus ponens, in all standard proof systems), $\Gamma \cup \{\neg\varphi\} \vdash \bot$.

Now assume $\Gamma \cup \{\neg\varphi\}$ is inconsistent. By the deduction theorem (THM-DED001, proved for each system in §4.7 below), $\Gamma \vdash \neg\varphi \to \bot$. Since $(\neg\varphi \to \bot) \to \neg\neg\varphi$ and $\neg\neg\varphi \to \varphi$ are derivable, we obtain $\Gamma \vdash \varphi$ by modus ponens. □

Proposition 4.19. *If $\Gamma \vdash \varphi$ and $\neg\varphi \in \Gamma$, then Γ is inconsistent.*

Proof. Since $\neg\varphi \in \Gamma$, by reflexivity $\Gamma \vdash \neg\varphi$. Together with $\Gamma \vdash \varphi$ and the derivability of $\neg\varphi \to (\varphi \to \bot)$, two applications of modus ponens yield $\Gamma \vdash \bot$. □

Proposition 4.20. *If $\Gamma \cup \{\varphi\}$ and $\Gamma \cup \{\neg\varphi\}$ are both inconsistent, then Γ is inconsistent.*

Proof. By Proposition 4.18, $\Gamma \cup \{\neg\varphi\}$ inconsistent implies $\Gamma \vdash \varphi$. Since $\Gamma \cup \{\varphi\}$ is also inconsistent, Proposition 4.17 gives that Γ is inconsistent. □

4.1.8 Derived and Admissible Rules

Definition 4.21 (Derived Rule)**.** A rule $\frac{\varphi_1 \cdots \varphi_n}{\psi}$ is a *derived rule* of a proof system if there exists a derivation of ψ from assumptions $\varphi_1, \ldots, \varphi_n$ using only the primitive rules. Derived rules serve as shortcuts that do not extend the system's deductive power.

Definition 4.22 (Admissible Rule)**.** A rule $\frac{\varphi_1 \cdots \varphi_n}{\psi}$ is *admissible* if: whenever $\vdash \varphi_1, \ldots, \vdash \varphi_n$ are all provable (as theorems), then $\vdash \psi$ is also provable. Unlike derived rules, admissible rules need not yield a derivation using the premises directly.

Remark 17. Every derived rule is admissible, but not conversely. For example, the cut rule is admissible in the sequent calculus **LK** (this is the content of the cut-elimination theorem, see CP-010, Cut Elimination, §4.4), but in a system with cut it is a primitive (hence trivially derived) rule. The distinction matters for proof search: derived rules can always be "compiled away" by inlining their justifying derivation, while admissible rules may require a global transformation of the proof.

4.1.9 Deductive Closure and Conservative Extension

Definition 4.23 (Deductive Closure)**.** The *deductive closure* of a set of formulas Γ is the set

$$\mathrm{Thm}(\Gamma) = \{\varphi : \Gamma \vdash \varphi\}.$$

A set Γ is *deductively closed* if $\Gamma = \mathrm{Thm}(\Gamma)$.

Definition 4.24 (Conservative Extension)**.** A theory T' in language $\mathcal{L}' \supseteq \mathcal{L}$ is a *conservative extension* of a theory T in $\mathcal{L}$ if for every $\mathcal{L}$-sentence φ: $T' \vdash \varphi$ implies $T \vdash \varphi$.

Remark 18. Conservative extensions are important in mathematical logic because they guarantee that expanding a theory with new symbols and axioms does not prove new theorems in the original language. This notion appears throughout the metatheory: definitional extensions are conservative, and the method of Henkin constants used in the completeness proof (see §6.3) produces a conservative extension of the original theory.

4.1.10 Maximally Consistent Sets

Definition 4.25 (Maximally Consistent Set)**.** A set Γ of sentences is *maximally consistent* iff

1. Γ is consistent, and
2. if $\Gamma \subsetneq \Gamma'$, then Γ' is inconsistent.

Equivalently, Γ is maximally consistent iff Γ is consistent and for every sentence φ: if $\Gamma \cup \{\varphi\}$ is consistent, then $\varphi \in \Gamma$.

Maximally consistent sets are central to the completeness proof. Every consistent set of sentences is contained in a maximally consistent set (by Lindenbaum's Lemma, see THM-DED005, §4.7). A maximally consistent set Γ contains, for each sentence φ, either φ or $\neg\varphi$. This property is what allows us to construct a structure satisfying Γ in the completeness argument.

Proposition 4.26. *Suppose Γ is maximally consistent. Then:*

1. *If $\Gamma \vdash \varphi$, then $\varphi \in \Gamma$.*
2. *For any φ, either $\varphi \in \Gamma$ or $\neg\varphi \in \Gamma$.*
3. *$(\varphi \wedge \psi) \in \Gamma$ iff both $\varphi \in \Gamma$ and $\psi \in \Gamma$.*
4. *$(\varphi \vee \psi) \in \Gamma$ iff either $\varphi \in \Gamma$ or $\psi \in \Gamma$.*
5. *$(\varphi \to \psi) \in \Gamma$ iff either $\varphi \notin \Gamma$ or $\psi \in \Gamma$.*

Proof. Let Γ be maximally consistent throughout.

1. If $\Gamma \vdash \varphi$ and $\varphi \notin \Gamma$, then since Γ is maximally consistent, $\Gamma \cup \{\varphi\}$ is inconsistent. By Proposition 4.17, Γ is inconsistent, contradicting the hypothesis. Hence $\varphi \in \Gamma$.
2. Suppose both $\varphi \notin \Gamma$ and $\neg\varphi \notin \Gamma$. Then $\Gamma \cup \{\varphi\}$ and $\Gamma \cup \{\neg\varphi\}$ are both inconsistent. By Proposition 4.20, Γ is inconsistent—a contradiction.
3. For the forward direction: if $(\varphi \wedge \psi) \in \Gamma$ then $\Gamma \vdash \varphi \wedge \psi$. Since $\varphi \wedge \psi \vdash \varphi$ and $\varphi \wedge \psi \vdash \psi$, we get $\Gamma \vdash \varphi$ and $\Gamma \vdash \psi$ by transitivity. By (1), $\varphi \in \Gamma$ and $\psi \in \Gamma$. For the reverse: if $\varphi, \psi \in \Gamma$ then $\Gamma \vdash \varphi$ and $\Gamma \vdash \psi$, so $\Gamma \vdash \varphi \wedge \psi$, and by (1), $(\varphi \wedge \psi) \in \Gamma$.
4. Analogous, using the derivability properties of $\vee$.
5. Analogous, using the derivability properties of $\to$.

□

4.1.11 Generic Connective and Quantifier Derivability

The following propositions state basic derivability facts for the propositional connectives and quantifiers that hold in all standard proof systems. They are used in the proof of the completeness theorem. System-specific derivations establishing these facts are given in DED.2–DED.5.

Proposition 4.27. *1. Both $\varphi \wedge \psi \vdash \varphi$ and $\varphi \wedge \psi \vdash \psi$.*

2. $\varphi, \psi \vdash \varphi \wedge \psi$.

The analogous facts hold for disjunction:

Proposition 4.28. *1. $\varphi \vee \psi, \neg\varphi, \neg\psi$ is inconsistent.*

2. Both $\varphi \vdash \varphi \vee \psi$ and $\psi \vdash \varphi \vee \psi$.

Proposition 4.29. *1. $\varphi, \varphi \rightarrow \psi \vdash \psi$.*

2. Both $\neg\varphi \vdash \varphi \rightarrow \psi$ and $\psi \vdash \varphi \rightarrow \psi$.

Proof sketch. (1) In axiomatic deduction, modus ponens applied to φ and the axiom $\varphi \rightarrow \psi$ immediately yields ψ. In natural deduction, $\rightarrow$-elimination serves the same role. In the sequent calculus, Left-$\rightarrow$ gives ψ from $\varphi \rightarrow \psi$ and φ. (2) From ψ we derive $\varphi \rightarrow \psi$ by axiom (A7) (or $\rightarrow$-introduction / Right-$\rightarrow$); from $\neg\varphi$ we derive $\varphi \rightarrow \psi$ by axiom (A10) (or analogous rules). Detailed derivations appear in DED.2 (axiomatic), DED.3 (natural deduction), DED.4 (sequent calculus), and DED.5 (tableaux). □

Quantifiers interact with derivability through a generalization principle:

Theorem 4.30 (Strong Generalization)**.** *If c is a constant symbol not occurring in Γ or $\varphi(x)$ and $\Gamma \vdash \varphi(c)$, then $\Gamma \vdash \forall x\, \varphi(x)$.*

Proof. By the deduction theorem (THM-DED001), $\Gamma \vdash \top \rightarrow \varphi(c)$. Since c does not occur in Γ or $\top$, the quantifier rule (or its equivalent in other systems) gives $\Gamma \vdash \top \rightarrow \forall x\, \varphi(x)$. By the deduction theorem again, $\Gamma \vdash \forall x\, \varphi(x)$. □

The basic quantifier derivability facts needed for completeness are:

Proposition 4.31. *1. $\varphi(t) \vdash \exists x\, \varphi(x)$.*

2. $\forall x\, \varphi(x) \vdash \varphi(t)$.

Proof. Both follow from the quantifier axioms (or quantifier rules, in natural deduction and the sequent calculus) and the deduction theorem (or the corresponding introduction/elimination rules). For detailed derivations, see DED.2–DED.5. □

4.2 Axiomatic (Hilbert) Systems

The axiomatic system instantiates the generic proof-theoretic framework of §4.1 as follows. A derivation (4.5) is a finite *sequence* of sentences; the system has many axiom schemas (4.1) and only two rules of inference (4.3): modus ponens and a quantifier rule. Provability (4.9) and consistency (4.16) are defined exactly as in §4.1; all structural properties established there (reflexivity, monotonicity, transitivity, compactness) hold with derivations understood as sequences.

Axiomatic derivation systems were introduced by Gottlob Frege in 1879, refined by Whitehead and Russell, and perfected by Hilbert and his students in the 1920s. Because derivations have a very simple structure, it is relatively easy to prove things *about* them (e.g., the deduction theorem), though finding derivations in practice is difficult.

4.2.1 Propositional Axioms and Modus Ponens

Definition 4.32 (Axiomatic System)**.** The *axiomatic (Hilbert-style) deduction system* is defined by the propositional axiom schemas of 4.33 below, the quantifier axioms and quantifier rule of §4.2.2, and the rule of modus ponens (4.34). A derivation from a set of sentences Γ is a finite sequence $\psi_1, \ldots, \psi_n$ in which every ψ_i is either

1. an element of Γ, or
2. an instance of one of the axiom schemas, or
3. justified by a rule of inference applied to earlier items in the sequence.

We write $\Gamma \vdash \varphi$ if there exists such a sequence ending in φ.

Definition 4.33 (Propositional Axioms)**.** The set Ax_0 of *axioms* for the propositional connectives comprises all formulas of the following forms:

$$(\varphi \wedge \psi) \to \varphi \tag{A1}$$
$$(\varphi \wedge \psi) \to \psi \tag{A2}$$
$$\varphi \to (\psi \to (\varphi \wedge \psi)) \tag{A3}$$
$$\varphi \to (\varphi \vee \psi) \tag{A4}$$
$$\varphi \to (\psi \vee \varphi) \tag{A5}$$
$$(\varphi \to \chi) \to ((\psi \to \chi) \to ((\varphi \vee \psi) \to \chi)) \tag{A6}$$
$$\varphi \to (\psi \to \varphi) \tag{A7}$$
$$(\varphi \to (\psi \to \chi)) \to ((\varphi \to \psi) \to (\varphi \to \chi)) \tag{A8}$$
$$(\varphi \to \psi) \to ((\varphi \to \neg\psi) \to \neg\varphi) \tag{A9}$$
$$\neg\varphi \to (\varphi \to \psi) \tag{A10}$$
$$\top \tag{A11}$$
$$\bot \to \varphi \tag{A12}$$
$$(\varphi \to \bot) \to \neg\varphi \tag{A13}$$
$$\neg\neg\varphi \to \varphi \tag{A14}$$

Definition 4.34 (Modus Ponens)**.** If ψ and $\psi \to \varphi$ already occur in a derivation, then φ is a correct inference step. We abbreviate this rule as MP.

Example 4.35. Here is an axiomatic derivation of $\vdash p \to p$:

1.	$p \to ((p \to p) \to p)$	(A7)
2.	$(p \to ((p \to p) \to p)) \to ((p \to (p \to p)) \to (p \to p))$	(A8)
3.	$(p \to (p \to p)) \to (p \to p)$	MP, 1, 2
4.	$p \to (p \to p)$	(A7)
5.	$p \to p$	MP, 4, 3

Even this simple identity requires five lines, illustrating why axiomatic systems are easy to reason *about* but hard to use directly.

4.2.2 Quantifier Axioms and Rules

Definition 4.36 (Quantifier Axioms)**.** The *axioms* governing quantifiers are all instances of:

$$\forall x\, \psi \to \psi(t), \tag{Q1}$$
$$\psi(t) \to \exists x\, \psi, \tag{Q2}$$

for any closed term t.

Definition 4.37 (Quantifier Rule)**.** The quantifier rule QR has two forms:

1. If $\psi \to \varphi(a)$ already occurs in the derivation and a does not occur in Γ or ψ, then $\psi \to \forall x\, \varphi(x)$ is a correct inference step.
2. If $\varphi(a) \to \psi$ already occurs in the derivation and a does not occur in Γ or ψ, then $\exists x\, \varphi(x) \to \psi$ is a correct inference step.

4.2.3 The Deduction Theorem (Axiomatic Proof)

The deduction theorem (4.77) is the central metatheorem for axiomatic systems. Below we give its full proof, including the quantifier-rule case.

Proposition 4.38 (Meta-Modus Ponens)**.** *If $\Gamma \vdash \varphi$ and $\Gamma \vdash \varphi \to \psi$, then $\Gamma \vdash \psi$.*

Proof. We have that $\{\varphi, \varphi \to \psi\} \vdash \psi$:

1.	φ	Hyp.
2.	$\varphi \to \psi$	Hyp.
3.	ψ	1, 2, MP

By transitivity (4.13), $\Gamma \vdash \psi$. □

Theorem 4.39 (Deduction Theorem — Axiomatic Proof)**.** *$\Gamma \cup \{\varphi\} \vdash \psi$ if and only if $\Gamma \vdash \varphi \to \psi$.*

Proof. The "if" direction is immediate: if $\Gamma \vdash \varphi \to \psi$ then $\Gamma \cup \{\varphi\} \vdash \varphi \to \psi$ by monotonicity (4.12), $\Gamma \cup \{\varphi\} \vdash \varphi$ by reflexivity (4.11), and Proposition 4.38 gives $\Gamma \cup \{\varphi\} \vdash \psi$.

For the "only if" direction, we proceed by induction on the length of the derivation of ψ from $\Gamma \cup \{\varphi\}$.

Base case. A derivation of length 1 consists of ψ alone, justified because $\psi \in \Gamma \cup \{\varphi\}$ or ψ is an axiom. If $\psi \in \Gamma$ or ψ is an axiom, then $\Gamma \vdash \psi$. Since $\Gamma \vdash \psi \to (\varphi \to \psi)$ by axiom (A7), Proposition 4.38 gives $\Gamma \vdash \varphi \to \psi$. If $\psi \equiv \varphi$, then $\Gamma \vdash \varphi \to \varphi$ is derivable using axioms (A7) and (A8) and two applications of MP.

Inductive step (modus ponens). Suppose the derivation of ψ from $\Gamma \cup \{\varphi\}$ ends with a step justified by modus ponens from earlier lines $\chi \to \psi$ and χ.

Then $\Gamma \cup \{\varphi\} \vdash \chi \to \psi$ and $\Gamma \cup \{\varphi\} \vdash \chi$, and both derivations are shorter. By induction hypothesis:

$$\Gamma \vdash \varphi \to (\chi \to \psi);$$
$$\Gamma \vdash \varphi \to \chi.$$

By axiom (A8),

$$\Gamma \vdash (\varphi \to (\chi \to \psi)) \to ((\varphi \to \chi) \to (\varphi \to \psi)),$$

and two applications of Proposition 4.38 give $\Gamma \vdash \varphi \to \psi$.

Inductive step (QR, *universal case).* Suppose $\psi \equiv \chi \to \forall x\, \theta(x)$ and a formula $\chi \to \theta(a)$ appears earlier in the derivation, where a does not occur in χ, φ, or Γ. By induction hypothesis, $\Gamma \vdash \varphi \to (\chi \to \theta(a))$. From

$$\vdash (\varphi \to (\chi \to \theta(a))) \to ((\varphi \wedge \chi) \to \theta(a))$$

and modus ponens we get $\Gamma \vdash (\varphi \wedge \chi) \to \theta(a)$. Since the eigenvariable condition still holds, a step justified by QR gives $\Gamma \vdash (\varphi \wedge \chi) \to \forall x\, \theta(x)$. From

$$\vdash ((\varphi \wedge \chi) \to \forall x\, \theta(x)) \to (\varphi \to (\chi \to \forall x\, \theta(x))),$$

modus ponens yields $\Gamma \vdash \varphi \to (\chi \to \forall x\, \theta(x))$, i.e., $\Gamma \vdash \varphi \to \psi$.

The existential case ($\psi \equiv \exists x\, \theta(x) \to \chi$) is symmetric: one replaces $\chi \to \theta(a)$ with $\theta(a) \to \chi$ throughout and applies the second form of QR. □

Notice how axioms (A7) and (A8) were chosen precisely so that the Deduction Theorem would hold.

4.2.4 Derivability Properties

The following propositions provide the derivability facts stated generically in §4.1, now with explicit axiomatic proofs.

Proposition 4.40. *1. Both $\varphi \wedge \psi \vdash \varphi$ and $\varphi \wedge \psi \vdash \psi$.*

2. $\varphi, \psi \vdash \varphi \wedge \psi$.

Proof. (1) From axioms (A1) and (A2) by modus ponens. (2) From axiom (A3) by two applications of modus ponens. □

Proposition 4.41. *1. $\varphi \lor \psi, \neg\varphi, \neg\psi$ is inconsistent.*

2. Both $\varphi \vdash \varphi \lor \psi$ and $\psi \vdash \varphi \lor \psi$.

Proof. (1) From axiom (A10) we derive $\{\neg\varphi\} \vdash \varphi \rightarrow \bot$ and $\{\neg\psi\} \vdash \psi \rightarrow \bot$. By axiom (A6) and the deduction theorem, $\{\varphi \lor \psi, \neg\varphi, \neg\psi\} \vdash \bot$. (2) From axioms (A4) and (A5) by modus ponens. □

Proposition 4.42. *1. $\varphi, \varphi \rightarrow \psi \vdash \psi$.*

2. Both $\neg\varphi \vdash \varphi \rightarrow \psi$ and $\psi \vdash \varphi \rightarrow \psi$.

Proof. (1) Immediate from modus ponens. (2) By axiom (A10) and axiom (A7), respectively, together with the deduction theorem. □

Theorem 4.43 (Strong Generalization)**.** *If c is a constant symbol not occurring in Γ or $\varphi(x)$ and $\Gamma \vdash \varphi(c)$, then $\Gamma \vdash \forall x\, \varphi(x)$.*

Proof. By the deduction theorem, $\Gamma \vdash \top \rightarrow \varphi(c)$. Since c does not occur in Γ or $\top$, the quantifier rule gives $\Gamma \vdash \top \rightarrow \forall x\, \varphi(x)$. By the deduction theorem again, $\Gamma \vdash \forall x\, \varphi(x)$. □

Proposition 4.44. *1. $\varphi(t) \vdash \exists x\, \varphi(x)$.*

2. $\forall x\, \varphi(x) \vdash \varphi(t)$.

Proof. (1) By axiom (Q2) and modus ponens. (2) By axiom (Q1) and modus ponens. □

4.2.5 Soundness

Proposition 4.45. *If φ is an axiom (propositional, quantifier, or identity), then $\mathfrak{M}, s \vDash \varphi$ for each structure $\mathfrak{M}$ and assignment s.*

Proof. We verify that all the axioms are valid. For instance, here is the case for axiom (Q1): suppose t is free for x in φ, and assume $\mathfrak{M}, s \vDash \forall x\, \varphi$. Then for each $s' \sim_x s$, also $\mathfrak{M}, s' \vDash \varphi$, and in particular this holds when $s'(x) = \mathrm{Val}_s^{\mathfrak{M}}(t)$. By the substitution lemma (see SYN.4), $\mathfrak{M}, s \vDash \varphi[t/x]$. This shows that $\mathfrak{M}, s \vDash (\forall x\, \varphi \rightarrow \varphi[t/x])$. The remaining propositional axioms are verified by truth-value analysis.

For the identity axioms: $t = t$ is valid since $\mathrm{Val}^{\mathfrak{M}}(t) = \mathrm{Val}^{\mathfrak{M}}(t)$. The axiom $t_1 = t_2 \to (\psi(t_1) \to \psi(t_2))$ is valid because if $\mathrm{Val}^{\mathfrak{M}}(t_1) = \mathrm{Val}^{\mathfrak{M}}(t_2)$, then by the substitution lemma $\mathfrak{M} \vDash \psi(t_1)$ iff $\mathfrak{M} \vDash \psi(t_2)$. □

Theorem 4.46 (Soundness)**.** *If $\Gamma \vdash \varphi$ then $\Gamma \vDash \varphi$.*

Proof. By induction on the length of the derivation of φ from Γ.

If there are no steps justified by inferences, then all formulas in the derivation are either axiom instances or in Γ. By Proposition 4.45, all axioms are valid, so if φ is an axiom then $\Gamma \vDash \varphi$. If $\varphi \in \Gamma$, then trivially $\Gamma \vDash \varphi$.

If the last step is justified by modus ponens, then there are formulas ψ and $\psi \to \varphi$ in the derivation, and the induction hypothesis applies to the parts of the derivation ending in those formulas (since they contain at least one fewer inference step). So by induction hypothesis, $\Gamma \vDash \psi$ and $\Gamma \vDash \psi \to \varphi$. Then $\Gamma \vDash \varphi$ by the semantic deduction theorem (see THM-SEM, Semantic Deduction, §SEM).

If the last step is justified by QR and has the form $\chi \to \forall x\, \psi(x)$, then there is a preceding step $\chi \to \psi(c)$ with c not in Γ, χ, or $\forall x\, \psi(x)$. By induction hypothesis, $\Gamma \vDash \chi \to \psi(c)$. By the semantic deduction theorem, $\Gamma \cup \{\chi\} \vDash \psi(c)$. Consider a structure $\mathfrak{M}$ with $\mathfrak{M} \vDash \Gamma \cup \{\chi\}$. We must show $\mathfrak{M} \vDash \forall x\, \psi(x)$, i.e., for every variable assignment s, $\mathfrak{M}, s \vDash \psi(x)$. Since $\Gamma \cup \{\chi\}$ consists of sentences, $\mathfrak{M}, s \vDash \theta$ for all $\theta \in \Gamma \cup \{\chi\}$. Let $\mathfrak{M}'$ be like $\mathfrak{M}$ except $c^{\mathfrak{M}'} = s(x)$. Since c does not occur in Γ or χ, $\mathfrak{M}' \vDash \Gamma \cup \{\chi\}$. Since $\Gamma \cup \{\chi\} \vDash \psi(c)$, $\mathfrak{M}' \vDash \psi(c)$. Since $\psi(c)$ is a sentence, $\mathfrak{M}', s \vDash \psi(c)$. By the substitution lemma, $\mathfrak{M}', s \vDash \psi(x)$. Since c does not occur in $\psi(x)$, $\mathfrak{M}, s \vDash \psi(x)$. Since s was arbitrary, $\mathfrak{M} \vDash \forall x\, \psi(x)$, and by the semantic deduction theorem, $\Gamma \vDash \chi \to \forall x\, \psi(x)$.

The case where the last step is $\exists x\, \psi(x) \to \chi$ is symmetric.

For the identity axioms, Proposition 4.45 ensures they are valid, so the base case of the induction applies. □

Corollary 4.47 (Weak Soundness)**.** *If $\vdash \varphi$, then φ is valid.*

Corollary 4.48. *If Γ is satisfiable, then it is consistent.*

Proof. We prove the contrapositive. If Γ is inconsistent, then $\Gamma \vdash \bot$. By Theorem 4.46, any structure $\mathfrak{M}$ satisfying Γ must satisfy $\bot$. Since $\mathfrak{M} \nvDash \bot$ for every $\mathfrak{M}$, no structure can satisfy Γ, so Γ is unsatisfiable. □

4.3 Natural Deduction

Natural deduction instantiates the generic framework of §4.1 differently from axiomatic deduction. There are no logical axioms; instead, each connective and quantifier has introduction and elimination rules, corresponding to natural inference patterns (e.g., conditional proof, proof by cases, indirect proof). A derivation (4.5) is a finite *tree* of sentences rather than a sequence. Provability (4.9) and consistency (4.16) are defined exactly as in §4.1.

A distinguishing feature of natural deduction is the mechanism of *assumption discharge* (4.8): certain rules cancel (*discharge*) hypothetical assumptions, so that they no longer count among the open assumptions of the derivation. The undischarged assumptions of a completed derivation are the set Γ from which the conclusion is derived.

Natural deduction systems were developed by Gentzen and Jaśkowski in the 1930s, and later refined by Prawitz and Fitch. In the philosophy of logic, the rules of natural deduction have sometimes been taken to define the meanings of the logical operators ("proof-theoretic semantics").

4.3.1 Rules and Derivations

Definition 4.49 (Assumption)**.** An *assumption* is any sentence in the topmost position of any branch of a derivation tree.

Derivations in natural deduction are certain trees of sentences, where the topmost sentences are assumptions, and if a sentence stands below one, two, or three other sentences, it must follow correctly by a rule of inference. The sentences at the top of the inference are the *premises* and the sentence below the *conclusion*. The rules come in pairs—an introduction and an elimination rule for each operator—and some rules allow an assumption to be *discharged*. To indicate which assumption is discharged by which inference, both receive a matching label: the assumption is written "$[\varphi]^n$."

It is customary to consider rules for all the operators $\land$, $\lor$, $\to$, $\neg$, and $\bot$, even if some of those are defined.

4.3.2 Propositional Rules

Rules for $\land$

$$\frac{\varphi \qquad \psi}{\varphi \land \psi}\ \land\text{Intro}$$

$$\frac{\varphi \land \psi}{\varphi}\ \land\text{Elim}$$

$$\frac{\varphi \land \psi}{\psi}\ \land\text{Elim}$$

Rules for $\lor$

$$\frac{\varphi}{\varphi \lor \psi}\lor\text{Intro} \qquad \frac{\psi}{\varphi \lor \psi}\lor\text{Intro}$$

$$ {}_{n}\frac{\varphi \lor \psi \qquad \begin{matrix}[\varphi]^n\\ \chi\end{matrix} \qquad \begin{matrix}[\psi]^n\\ \chi\end{matrix}}{\chi}\lor\text{Elim}$$

Rules for $\to$

$$ {}_{n}\frac{\begin{matrix}[\varphi]^n\\ \psi\end{matrix}}{\varphi \to \psi}\to\text{Intro} \qquad \frac{\varphi \to \psi \qquad \varphi}{\psi}\to\text{Elim}$$

Rules for $\neg$

$$ {}_{n}\frac{\begin{matrix}[\varphi]^n\\ \bot\end{matrix}}{\neg\varphi}\neg\text{Intro} \qquad \frac{\neg\varphi \qquad \varphi}{\bot}\neg\text{Elim}$$

Rules for $\bot$

$$\frac{\bot}{\varphi}\bot_I \qquad {}_{n}\frac{\begin{matrix}[\neg\varphi]^n\\ \bot\end{matrix}}{\varphi}\bot_C$$

Note that $\neg$Intro and $\bot_C$ are very similar: the difference is that $\neg$Intro derives a negated sentence $\neg\varphi$ while $\bot_C$ derives a positive sentence φ.

Whenever a rule indicates that some assumption may be discharged, this is a permission, not a requirement. In the $\to$Intro rule, for instance, we may discharge any number of assumptions of the form φ, including zero.

4.3.3 Quantifier Rules

Rules for $\forall$

$$\frac{\varphi(a)}{\forall x\, \varphi(x)}\forall\text{Intro} \qquad \frac{\forall x\, \varphi(x)}{\varphi(t)}\forall\text{Elim}$$

In $\forall$Elim, t is a closed term. In $\forall$Intro, a is a constant symbol which does not occur in the conclusion $\forall x\, \varphi(x)$ or in any undischarged assumption. We call a the *eigenvariable* of the $\forall$Intro inference.

Rules for $\exists$

$$\frac{\varphi(t)}{\exists x\,\varphi(x)}\;\exists\text{Intro} \qquad\qquad {}_{n}\frac{\exists x\,\varphi(x) \qquad \begin{matrix}[\varphi(a)]^n\\ \chi\end{matrix}}{\chi}\;\exists\text{Elim}$$

Again, t is a closed term, and a is a constant symbol which does not occur in $\exists x\,\varphi(x)$, in χ, or in any undischarged assumption (other than $\varphi(a)$). We call a the *eigenvariable* of the $\exists$Elim inference.

The condition that an eigenvariable not occur in the premises or in any undischarged assumption is the *eigenvariable condition*. It is necessary to ensure soundness.

Rules for Identity

$$\frac{}{t = t}\;{=}\text{Intro} \qquad\qquad \frac{t_1 = t_2 \qquad \varphi(t_1)}{\varphi(t_2)}\;{=}\text{Elim} \qquad \frac{t_1 = t_2 \qquad \varphi(t_2)}{\varphi(t_1)}\;{=}\text{Elim}$$

In the above rules, t, t_1, and t_2 are closed terms. The $=$Intro rule derives $t = t$ from no assumptions.

Example 4.50. As a first example, here is a derivation of $p \to p$ from no undischarged assumptions:

$${}_{1}\frac{[p]^1}{p \to p}\;{\to}\text{Intro}$$

The sole leaf is the assumption p, discharged by $\to$Intro at label 1. Since no assumptions remain open, this is a derivation of $p \to p$ from $\emptyset$.

4.3.4 Derivations

Definition 4.51 (Derivation)**.** A *derivation* of a sentence φ from assumptions Γ is a finite tree of sentences satisfying:

1. The topmost sentences are either in Γ or are discharged by an inference in the tree.
2. The bottommost sentence is φ.
3. Every sentence except φ is a premise of a correct application of an inference rule whose conclusion stands directly below it.

We call φ the *conclusion* and Γ the *undischarged assumptions*. If such a derivation exists, we write $\Gamma \vdash \varphi$. If every assumption is discharged, we write $\vdash \varphi$.

4.3.5 Soundness

Theorem 4.52 (Soundness). *If φ is derivable from the undischarged assumptions Γ, then $\Gamma \vDash \varphi$.*

Proof. Let δ be a derivation of φ. We proceed by induction on the number of inferences in δ.

Base case. If there are no inferences, δ consists of a single sentence φ that is an undischarged assumption. Any structure satisfying Γ satisfies φ since $\varphi \in \Gamma$.

Inductive step. Suppose δ contains n inferences. We assume the induction hypothesis for each sub-derivation (which has fewer than n inferences) and distinguish cases by the last inference.

1. $\neg$Intro*:* The derivation ends in

$$\begin{array}{c} \Gamma, [\varphi]^n \\ \delta_1 \\ n\ \dfrac{\bot}{\neg\varphi}\ \neg\text{Intro} \end{array}$$

   By induction hypothesis, $\Gamma \cup \{\varphi\} \vDash \bot$. Suppose $\mathfrak{M} \vDash \Gamma$ but $\mathfrak{M} \nvDash \neg\varphi$, i.e., $\mathfrak{M} \vDash \varphi$. Then $\mathfrak{M} \vDash \Gamma \cup \{\varphi\}$, so $\mathfrak{M} \vDash \bot$, a contradiction. Hence $\mathfrak{M} \vDash \neg\varphi$.

2. $\wedge$Elim*:* By induction hypothesis $\Gamma \vDash \varphi \wedge \psi$; by definition of satisfaction, $\mathfrak{M} \vDash \varphi$ (or $\mathfrak{M} \vDash \psi$, for the other variant).

3. $\vee$Intro*:* By induction hypothesis $\Gamma \vDash \varphi$; since $\mathfrak{M} \vDash \varphi$ implies $\mathfrak{M} \vDash \varphi \vee \psi$.

4. $\rightarrow$Intro*:* By induction hypothesis $\Gamma \cup \{\varphi\} \vDash \psi$. Suppose $\mathfrak{M} \vDash \Gamma$ but $\mathfrak{M} \nvDash \varphi \rightarrow \psi$. Then $\mathfrak{M} \vDash \varphi$ and $\mathfrak{M} \nvDash \psi$, so $\mathfrak{M} \vDash \Gamma \cup \{\varphi\}$, giving $\mathfrak{M} \vDash \psi$, contradiction.

5. $\rightarrow$Elim*:* By induction hypothesis $\Gamma_1 \vDash \varphi \rightarrow \psi$ and $\Gamma_2 \vDash \varphi$. If $\mathfrak{M} \vDash \Gamma_1 \cup \Gamma_2$, then $\mathfrak{M} \vDash \varphi \rightarrow \psi$ and $\mathfrak{M} \vDash \varphi$, so $\mathfrak{M} \vDash \psi$.

6. $\neg$Elim*:* By induction hypothesis $\Gamma_1 \vDash \neg\varphi$ and $\Gamma_2 \vDash \varphi$. If $\mathfrak{M} \vDash \Gamma_1 \cup \Gamma_2$ then both $\mathfrak{M} \vDash \neg\varphi$ and $\mathfrak{M} \vDash \varphi$, giving $\mathfrak{M} \vDash \bot$.

7. $\bot_I$*:* By induction hypothesis $\Gamma \vDash \bot$. Since $\mathfrak{M} \nvDash \bot$ for every $\mathfrak{M}$, no structure satisfies Γ, so $\Gamma \vDash \varphi$ vacuously.

8. $\bot_C$*:* By induction hypothesis $\Gamma \cup \{\neg\varphi\} \vDash \bot$. Suppose $\mathfrak{M} \vDash \Gamma$ but $\mathfrak{M} \nvDash \varphi$, i.e., $\mathfrak{M} \vDash \neg\varphi$. Then $\mathfrak{M} \vDash \Gamma \cup \{\neg\varphi\}$, so $\mathfrak{M} \vDash \bot$, contradiction.

9. $\wedge$Intro*:* By induction hypothesis $\Gamma_1 \vDash \varphi$ and $\Gamma_2 \vDash \psi$. If $\mathfrak{M} \vDash \Gamma_1 \cup \Gamma_2$, then $\mathfrak{M} \vDash \varphi$ and $\mathfrak{M} \vDash \psi$, so $\mathfrak{M} \vDash \varphi \wedge \psi$.

10. $\vee$Elim*:* By induction hypothesis $\Gamma_1 \vDash \varphi \vee \psi$, $\Gamma_2 \cup \{\varphi\} \vDash \chi$, and $\Gamma_3 \cup \{\psi\} \vDash \chi$. If $\mathfrak{M} \vDash \Gamma_1 \cup \Gamma_2 \cup \Gamma_3$, then $\mathfrak{M} \vDash \varphi \vee \psi$, so either $\mathfrak{M} \vDash \varphi$ or $\mathfrak{M} \vDash \psi$; in either case $\mathfrak{M} \vDash \chi$.

11. $\forall$Intro*:* By induction hypothesis $\Gamma \vDash \varphi(a)$ where a does not occur in Γ or $\forall x\, \varphi(x)$. Suppose $\mathfrak{M} \vDash \Gamma$. We must show $\mathfrak{M} \vDash \forall x\, \varphi(x)$, i.e., for every variable assignment s, $\mathfrak{M}, s \vDash \varphi(x)$. Let $\mathfrak{M}'$ be like $\mathfrak{M}$ except $a^{\mathfrak{M}'} = s(x)$. Since a does not occur in Γ, $\mathfrak{M}' \vDash \Gamma$, so $\mathfrak{M}' \vDash \varphi(a)$, whence $\mathfrak{M}', s \vDash \varphi(a)$. By the substitution lemma, $\mathfrak{M}', s \vDash \varphi(x)$. Since a does not occur in $\varphi(x)$, $\mathfrak{M}, s \vDash \varphi(x)$.

12. $\exists$Intro*:* By induction hypothesis $\Gamma \vDash \varphi(t)$. By the substitution lemma, $\mathfrak{M} \vDash \varphi(t)$ implies $\mathfrak{M} \vDash \exists x\, \varphi(x)$.

13. $\forall$Elim*:* By induction hypothesis $\Gamma \vDash \forall x\, \varphi(x)$. If $\mathfrak{M} \vDash \forall x\, \varphi(x)$, by the substitution lemma $\mathfrak{M} \vDash \varphi(t)$.

14. $\exists$Elim*:* By induction hypothesis $\Gamma_1 \vDash \exists x\, \varphi(x)$ and $\Gamma_2 \cup \{\varphi(a)\} \vDash \chi$, where a does not occur in $\exists x\, \varphi(x)$, χ, or the undischarged assumptions. The argument is analogous to the $\forall$Intro case, constructing a suitable $\mathfrak{M}'$.

15. $=$Intro*:* $t = t$ is valid since $\mathrm{Val}^{\mathfrak{M}}(t) = \mathrm{Val}^{\mathfrak{M}}(t)$ for every $\mathfrak{M}$.

16. $=$Elim*:* By induction hypothesis $\Gamma_1 \vDash t_1 = t_2$ and $\Gamma_2 \vDash \varphi(t_1)$. If $\mathfrak{M} \vDash \Gamma_1 \cup \Gamma_2$, then $\mathrm{Val}^{\mathfrak{M}}(t_1) = \mathrm{Val}^{\mathfrak{M}}(t_2)$. By the substitution lemma, $\mathfrak{M} \vDash \varphi(t_1)$ iff $\mathfrak{M} \vDash \varphi(t_2)$, so $\mathfrak{M} \vDash \varphi(t_2)$.

□

Corollary 4.53 (Weak Soundness)**.** *If* $\vdash \varphi$*, then* φ *is valid.*

Corollary 4.54. *If* Γ *is satisfiable, then it is consistent.*

Proof. Contrapositive: if Γ is inconsistent, then $\Gamma \vdash \bot$. By Theorem 4.52, any structure satisfying Γ must satisfy $\bot$. Since $\mathfrak{M} \nvDash \bot$ for every $\mathfrak{M}$, Γ is unsatisfiable. □

Remark 19. The structural properties of §4.1 (reflexivity, monotonicity, transitivity, compactness) hold for natural deduction. Transitivity uses implication-introduction and elimination rather than sequence concatenation.

4.4 Sequent Calculus

The sequent calculus (Gentzen's **LK**) instantiates the generic framework of §4.1 with derivations that are trees of *sequents* (4.6) rather than trees of single formulas. The system has no axiom schemas in the Hilbert sense; instead, it employs initial sequents, logical rules (one left and one right rule per connective), and the structural rules of §4.1.3 (4.7). Provability (4.9) and consistency (4.16) are defined as before: $\Gamma \vdash \varphi$ iff for some finite $\Gamma_0 \subseteq \Gamma$, the sequent $\Gamma_0 \Rightarrow \varphi$ has a derivation.

4.4.1 Sequents and Initial Sequents

For the following, let $\Gamma, \Delta, \Pi, \Lambda$ represent finite sequences of sentences.

Definition 4.55 (Sequent)**.** A *sequent* is an expression $\Gamma \Rightarrow \Delta$ where Γ (the *antecedent*) and Δ (the *succedent*) are finite (possibly empty) sequences of sentences.

The intuitive reading of $\Gamma \Rightarrow \Delta$ is: if all sentences in the antecedent hold, then at least one of the sentences in the succedent holds. That is, if $\Gamma = \langle \varphi_1, \ldots, \varphi_m \rangle$ and $\Delta = \langle \psi_1, \ldots, \psi_n \rangle$, then $\Gamma \Rightarrow \Delta$ corresponds to

$$(\varphi_1 \wedge \cdots \wedge \varphi_m) \rightarrow (\psi_1 \vee \cdots \vee \psi_n).$$

When Γ is empty, $\Rightarrow \Delta$ asserts $\psi_1 \vee \cdots \vee \psi_n$. When Δ is empty, $\Gamma \Rightarrow$ asserts $\neg(\varphi_1 \wedge \cdots \wedge \varphi_m)$.

If Γ is a sequence, we write Γ, φ for the result of appending φ to the right end of Γ (and φ, Γ for appending to the left). Γ, Δ denotes concatenation.

Definition 4.56 (Initial Sequent)**.** An *initial sequent* is a sequent of one of the following forms:

1. $\varphi \Rightarrow \varphi$ for any sentence φ;
2. $\Rightarrow \top$;
3. $\bot \Rightarrow$.

Derivations in the sequent calculus are trees of sequents, where the topmost sequents are initial sequents, and the logical rules are named for the main operator of the sentence they introduce. Each comes in two forms: a left rule (introducing the operator in the antecedent) and a right rule (introducing it in the succedent).

4.4.2 Propositional Rules

Rules for $\neg$

$$\frac{\Gamma \Rightarrow \Delta, \varphi}{\neg\varphi, \Gamma \Rightarrow \Delta}\neg\mathrm{L} \qquad \frac{\varphi, \Gamma \Rightarrow \Delta}{\Gamma \Rightarrow \Delta, \neg\varphi}\neg\mathrm{R}$$

Rules for $\wedge$

$$\frac{\varphi, \Gamma \Rightarrow \Delta}{\varphi \wedge \psi, \Gamma \Rightarrow \Delta}\wedge\mathrm{L} \qquad \frac{\psi, \Gamma \Rightarrow \Delta}{\varphi \wedge \psi, \Gamma \Rightarrow \Delta}\wedge\mathrm{L}$$

$$\frac{\Gamma \Rightarrow \Delta, \varphi \qquad \Gamma \Rightarrow \Delta, \psi}{\Gamma \Rightarrow \Delta, \varphi \wedge \psi}\wedge\mathrm{R}$$

Rules for $\vee$

$$\frac{\varphi, \Gamma \Rightarrow \Delta \qquad \psi, \Gamma \Rightarrow \Delta}{\varphi \vee \psi, \Gamma \Rightarrow \Delta}\vee\mathrm{L}$$

$$\frac{\Gamma \Rightarrow \Delta, \varphi}{\Gamma \Rightarrow \Delta, \varphi \vee \psi}\vee\mathrm{R} \qquad \frac{\Gamma \Rightarrow \Delta, \psi}{\Gamma \Rightarrow \Delta, \varphi \vee \psi}\vee\mathrm{R}$$

Rules for $\rightarrow$

$$\frac{\Gamma \Rightarrow \Delta, \varphi \qquad \psi, \Pi \Rightarrow \Lambda}{\varphi \rightarrow \psi, \Gamma, \Pi \Rightarrow \Delta, \Lambda}\rightarrow\mathrm{L} \qquad \frac{\varphi, \Gamma \Rightarrow \Delta, \psi}{\Gamma \Rightarrow \Delta, \varphi \rightarrow \psi}\rightarrow\mathrm{R}$$

4.4.3 Structural Rules

The structural rules (4.7) of **LK** allow rearranging sentences in the antecedent and succedent. Because the logical rules require that the principal sentence stand at a specific position, exchange is needed to move it there; weakening and contraction handle unused or duplicated formulas.

Weakening

$$\frac{\Gamma \Rightarrow \Delta}{\varphi, \Gamma \Rightarrow \Delta}\mathrm{WL} \qquad \frac{\Gamma \Rightarrow \Delta}{\Gamma \Rightarrow \Delta, \varphi}\mathrm{WR}$$

Contraction

$$\dfrac{\varphi, \varphi, \Gamma \Rightarrow \Delta}{\varphi, \Gamma \Rightarrow \Delta}\,\text{CL} \qquad \dfrac{\Gamma \Rightarrow \Delta, \varphi, \varphi}{\Gamma \Rightarrow \Delta, \varphi}\,\text{CR}$$

Exchange

$$\dfrac{\Gamma, \varphi, \psi, \Pi \Rightarrow \Delta}{\Gamma, \psi, \varphi, \Pi \Rightarrow \Delta}\,\text{XL} \qquad \dfrac{\Gamma \Rightarrow \Delta, \varphi, \psi, \Lambda}{\Gamma \Rightarrow \Delta, \psi, \varphi, \Lambda}\,\text{XR}$$

A series of weakening, contraction, and exchange inferences is often indicated by double inference lines.

Cut

$$\dfrac{\Gamma \Rightarrow \Delta, \varphi \qquad \varphi, \Pi \Rightarrow \Lambda}{\Gamma, \Pi \Rightarrow \Delta, \Lambda}\,\text{Cut}$$

The cut rule is not strictly necessary, but makes it considerably easier to reuse and combine derivations.

Remark 20 (Cut Elimination). *Cut Elimination (Gentzen's Hauptsatz):* The Cut rule is *admissible* in **LK**—every **LK**-derivation using Cut can be transformed into one without Cut. The proof is a detailed structural induction on the complexity of the cut formula and the height of the derivation; it is beyond our scope here. Cut elimination has profound consequences: it implies the subformula property (every formula in a cut-free derivation is a subformula of the end-sequent), which in turn yields consistency proofs, decidability results, and interpolation theorems.

4.4.4 Quantifier Rules

Rules for $\forall$

$$\dfrac{\varphi(t), \Gamma \Rightarrow \Delta}{\forall x\, \varphi(x), \Gamma \Rightarrow \Delta}\,\forall\text{L} \qquad \dfrac{\Gamma \Rightarrow \Delta, \varphi(a)}{\Gamma \Rightarrow \Delta, \forall x\, \varphi(x)}\,\forall\text{R}$$

In $\forall$L, t is a closed term. In $\forall$R, a is a constant symbol which must not occur in the lower sequent. We call a the *eigenvariable* of the $\forall$R inference.

Rules for $\exists$

$$\frac{\varphi(a), \Gamma \Rightarrow \Delta}{\exists x\, \varphi(x), \Gamma \Rightarrow \Delta}\ \exists\text{L} \qquad \frac{\Gamma \Rightarrow \Delta, \varphi(t)}{\Gamma \Rightarrow \Delta, \exists x\, \varphi(x)}\ \exists\text{R}$$

Again, t is a closed term, and a is a constant symbol not occurring in the lower sequent. The *eigenvariable condition* requires that the eigenvariable a not occur anywhere in the lower sequent of the $\forall$R or $\exists$L inference.

Identity Rules

Definition 4.57 (Initial Sequents for $=$)**.** If t is a closed term, then $\Rightarrow t = t$ is an initial sequent.

The rules for $=$ are (t_1 and t_2 are closed terms):

$$\frac{t_1 = t_2, \Gamma \Rightarrow \Delta, \varphi(t_1)}{t_1 = t_2, \Gamma \Rightarrow \Delta, \varphi(t_2)}\ = \qquad \frac{t_1 = t_2, \Gamma \Rightarrow \Delta, \varphi(t_2)}{t_1 = t_2, \Gamma \Rightarrow \Delta, \varphi(t_1)}\ =$$

Example 4.58. A derivation of the sequent $\Rightarrow p \to p$:

$$\frac{p \Rightarrow p}{\Rightarrow p \to p}\ \to\text{R}$$

The topmost sequent $p \Rightarrow p$ is an initial sequent. The $\to$R rule moves p from the antecedent into the conditional on the succedent side.

4.4.5 Derivations

Definition 4.59 (**LK**-Derivation)**.** An **LK**-*derivation* of a sequent S is a finite tree of sequents satisfying:

1. The topmost sequents are initial sequents.
2. The bottommost sequent is S.
3. Every sequent except S is a premise of a correct application of an inference rule whose conclusion stands directly below it.

We say S is the *end-sequent* and that S is **LK**-*derivable*.

4.4.6 Soundness

Definition 4.60 (Valid Sequent)**.** A structure $\mathfrak{M}$ *satisfies* a sequent $\Gamma \Rightarrow \Delta$ iff either $\mathfrak{M} \nvDash \varphi$ for some $\varphi \in \Gamma$ or $\mathfrak{M} \vDash \varphi$ for some $\varphi \in \Delta$. A sequent is *valid* iff every structure satisfies it.

Theorem 4.61 (Soundness)**.** *If* **LK** *derives $\Theta \Rightarrow \Xi$, then $\Theta \Rightarrow \Xi$ is valid.*

Proof. Let π be a derivation of $\Theta \Rightarrow \Xi$. We proceed by induction on the number of inferences n in π.

If $n = 0$, then π is an initial sequent. Every initial sequent $\varphi \Rightarrow \varphi$ is valid, since for every $\mathfrak{M}$, either $\mathfrak{M} \nvDash \varphi$ or $\mathfrak{M} \vDash \varphi$. The sequents $\Rightarrow \top$ and $\bot \Rightarrow$ are also valid. Identity initial sequents $\Rightarrow t = t$ are valid since $\mathrm{Val}^{\mathfrak{M}}(t) = \mathrm{Val}^{\mathfrak{M}}(t)$.

If $n > 0$, we distinguish cases by the last inference. By induction hypothesis the premise(s) are valid.

The cases fall into three groups: structural rules (weakening, contraction, exchange, cut), propositional rules ($\neg$, $\wedge$, $\vee$, $\rightarrow$), and quantifier rules ($\forall$, $\exists$). In each case the argument shows that validity is preserved from premise(s) to conclusion.

1. *Weakening:* If the premise $\Gamma \Rightarrow \Delta$ is valid, then so is $\varphi, \Gamma \Rightarrow \Delta$ (and $\Gamma \Rightarrow \Delta, \varphi$), since any witness to the validity of $\Gamma \Rightarrow \Delta$ also witnesses the validity of the weakened sequent.

2. $\neg$L*:* The premise is $\Gamma \Rightarrow \Delta, \varphi$ and the conclusion is $\neg\varphi, \Gamma \Rightarrow \Delta$. Given $\mathfrak{M}$, if $\mathfrak{M}$ falsifies some $\chi \in \Gamma$ or satisfies some $\chi \in \Delta$, the conclusion is satisfied. Otherwise, the validity of the premise forces $\mathfrak{M} \vDash \varphi$, whence $\mathfrak{M} \nvDash \neg\varphi$, and $\neg\varphi \in \Theta$ is falsified.

3. $\neg$R*:* Symmetric to the $\neg$L case.

4. $\wedge$L*:* The premise is $\varphi, \Gamma \Rightarrow \Delta$ and the conclusion is $\varphi \wedge \psi, \Gamma \Rightarrow \Delta$. If $\mathfrak{M} \nvDash \varphi$, then $\mathfrak{M} \nvDash \varphi \wedge \psi$, and the conclusion is satisfied. Otherwise $\mathfrak{M} \vDash \varphi$, and the validity of the premise gives the result. The case with ψ is analogous.

5. $\vee$R*:* The premise is $\Gamma \Rightarrow \Delta, \varphi$ and the conclusion is $\Gamma \Rightarrow \Delta, \varphi \vee \psi$. If $\mathfrak{M} \vDash \varphi$ then $\mathfrak{M} \vDash \varphi \vee \psi$. Otherwise the validity of the premise gives a witness in Γ or Δ.

6. $\rightarrow$R*:* The premise is $\varphi, \Gamma \Rightarrow \Delta, \psi$ and the conclusion is $\Gamma \Rightarrow \Delta, \varphi \rightarrow \psi$. If $\mathfrak{M} \nvDash \varphi$ or $\mathfrak{M} \vDash \psi$, then $\mathfrak{M} \vDash \varphi \rightarrow \psi$. Otherwise the validity of the premise gives a witness.

7. $\wedge$R*:* (Two premises.) If $\mathfrak{M}$ does not satisfy $\Gamma \Rightarrow \Delta$, then the validity of the first premise forces $\mathfrak{M} \vDash \varphi$ and the validity of the second forces $\mathfrak{M} \vDash \psi$, whence $\mathfrak{M} \vDash \varphi \wedge \psi$.

8. $\vee$L*:* (Two premises.) If $\mathfrak{M} \vDash \varphi \vee \psi$ then either $\mathfrak{M} \vDash \varphi$ or $\mathfrak{M} \vDash \psi$. In the former case the validity of the left premise gives the result; in the latter, the right premise.

9. $\rightarrow$L*:* (Two premises.) Suppose $\mathfrak{M}$ does not satisfy $\Gamma, \Pi \Rightarrow \Delta, \Lambda$. Then $\mathfrak{M}$ satisfies neither $\Gamma \Rightarrow \Delta$ nor $\Pi \Rightarrow \Lambda$. The validity of $\Gamma \Rightarrow \Delta, \varphi$ forces $\mathfrak{M} \vDash \varphi$, and the validity of $\psi, \Pi \Rightarrow \Lambda$ forces $\mathfrak{M} \nvDash \psi$. Hence $\mathfrak{M} \nvDash \varphi \rightarrow \psi$.

10. Cut*:* (Two premises.) Either $\mathfrak{M} \nvDash \varphi$ or $\mathfrak{M} \vDash \varphi$. In the first case, $\mathfrak{M}$ must satisfy $\Gamma \Rightarrow \Delta$ by the validity of the left premise. In the second, $\mathfrak{M}$ must satisfy $\Pi \Rightarrow \Lambda$ by the validity of the right premise.

11. $\forall$L*:* The premise is $\varphi(t), \Gamma \Rightarrow \Delta$ and the conclusion is $\forall x\, \varphi(x), \Gamma \Rightarrow \Delta$. If $\mathfrak{M} \vDash \forall x\, \varphi(x)$, then by the substitution lemma $\mathfrak{M} \vDash \varphi(t)$. The validity of the premise then gives the result. If $\mathfrak{M} \nvDash \forall x\, \varphi(x)$, the conclusion is satisfied since $\forall x\, \varphi(x)$ is in the antecedent.

12. $\forall$R*:* The premise is $\Gamma \Rightarrow \Delta, \varphi(a)$ where the eigenvariable condition holds. Suppose $\mathfrak{M} \vDash \Gamma$ and $\mathfrak{M} \nvDash \chi$ for all $\chi \in \Delta$. We must show $\mathfrak{M} \vDash \forall x\, \varphi(x)$, i.e., for all s, $\mathfrak{M}, s \vDash \varphi(x)$. Let $\mathfrak{M}'$ be like $\mathfrak{M}$ except $a^{\mathfrak{M}'} = s(x)$. Since a does not occur in Γ or Δ, the same truth values hold in $\mathfrak{M}'$. The validity of the premise then gives $\mathfrak{M}' \vDash \varphi(a)$. By the substitution lemma, $\mathfrak{M}', s \vDash \varphi(x)$, and since a does not occur in $\varphi(x)$, $\mathfrak{M}, s \vDash \varphi(x)$.

13. $\exists$L*:* Symmetric to the $\forall$R case.

14. $\exists$R*:* Symmetric to the $\forall$L case.

15. *Identity rule =:* The premise is $t_1 = t_2, \Gamma \Rightarrow \Delta, \varphi(t_1)$ and the conclusion is $t_1 = t_2, \Gamma \Rightarrow \Delta, \varphi(t_2)$. By induction hypothesis the premise is valid. If $\mathfrak{M} \vDash t_1 = t_2$ and $\mathfrak{M} \vDash \varphi(t_1)$, then $\mathrm{Val}^{\mathfrak{M}}(t_1) = \mathrm{Val}^{\mathfrak{M}}(t_2)$, and by the substitution lemma $\mathfrak{M} \vDash \varphi(t_2)$.

□

Corollary 4.62 (Weak Soundness)**.** *If* $\vdash \varphi$ *then* φ *is valid.*

Corollary 4.63. *If* $\Gamma \vdash \varphi$ *then* $\Gamma \vDash \varphi$*.*

Proof. If $\Gamma \vdash \varphi$ then for some finite $\Gamma_0 \subseteq \Gamma$, there is a derivation of $\Gamma_0 \Rightarrow \varphi$. By Theorem 4.61, every structure $\mathfrak{M}$ either falsifies some $\psi \in \Gamma_0$ or satisfies φ. Hence, if $\mathfrak{M} \vDash \Gamma$ then $\mathfrak{M} \vDash \varphi$. □

Corollary 4.64. *If Γ is satisfiable, then it is consistent.*

Proof. Contrapositive: if Γ is inconsistent, then there is a finite $\Gamma_0 \subseteq \Gamma$ and a derivation of $\Gamma_0 \Rightarrow$. By Theorem 4.61, $\Gamma_0 \Rightarrow$ is valid, i.e., for every $\mathfrak{M}$ there is $\chi \in \Gamma_0$ with $\mathfrak{M} \nvDash \chi$. Hence no structure satisfies Γ. □

Remark 21. In the sequent calculus, derivability is expressed via $\Gamma_0 \Rightarrow \varphi$. Transitivity corresponds to the Cut rule.

4.5 Tableaux

Tableaux instantiate the generic framework of §4.1 by working *refutationally*: to show $\Gamma \vdash \varphi$, one attempts to build a systematic survey of all ways the assumptions in Γ could be true and φ false, and demonstrates that every such possibility leads to a contradiction. Derivations (4.5) are finitely branching trees of *signed formulas* rather than trees of plain formulas or sequents. Provability (4.9) and consistency (4.16) are defined as in §4.1: $\Gamma \vdash \varphi$ iff $\{\mathbb{T}\psi_1, \dots, \mathbb{T}\psi_n, \mathbb{F}\varphi\}$ has a closed tableau for some $\psi_1, \dots, \psi_n \in \Gamma$.

4.5.1 Signed Formulas and Tableau Rules

Definition 4.65 (Signed Formula)**.** A *signed formula* is a pair consisting of a truth value sign and a sentence:

$$\mathbb{T}\varphi \quad \text{or} \quad \mathbb{F}\varphi.$$

Intuitively, $\mathbb{T}\varphi$ means "φ might be true" and $\mathbb{F}\varphi$ means "φ might be false" in some structure.

Each signed formula in a tableau is either an *assumption* (at the top) or obtained from a signed formula above it by a rule of inference. There are two rules per main operator—one for sign $\mathbb{T}$ and one for sign $\mathbb{F}$—and some rules branch the tree.

A branch is *closed* when it contains both $\mathbb{T}\varphi$ and $\mathbb{F}\varphi$. A *closed tableau* is one where every branch is closed. A closed tableau *for* φ is a closed tableau with root $\mathbb{F}\varphi$. If such a closed tableau exists, all possibilities for φ being false have been ruled out.

4.5.2 Propositional Rules

Rules for $\neg$

$$\frac{\mathbb{T}\neg\varphi}{\mathbb{F}\varphi}\;\neg\mathbb{T} \qquad\qquad \frac{\mathbb{F}\neg\varphi}{\mathbb{T}\varphi}\;\neg\mathbb{F}$$

Rules for $\wedge$

$$\frac{\mathbb{T}\varphi\wedge\psi}{\begin{array}{c}\mathbb{T}\varphi\\ \mathbb{T}\psi\end{array}}\;\wedge\mathbb{T} \qquad\qquad \frac{\mathbb{F}\varphi\wedge\psi}{\mathbb{F}\varphi \quad\mid\quad \mathbb{F}\psi}\;\wedge\mathbb{F}$$

Rules for $\vee$

$$\frac{\mathbb{T}\varphi\vee\psi}{\mathbb{T}\varphi \quad\mid\quad \mathbb{T}\psi}\;\vee\mathbb{T} \qquad\qquad \frac{\mathbb{F}\varphi\vee\psi}{\begin{array}{c}\mathbb{F}\varphi\\ \mathbb{F}\psi\end{array}}\;\vee\mathbb{F}$$

Rules for $\rightarrow$

$$\frac{\mathbb{T}\varphi\rightarrow\psi}{\mathbb{F}\varphi \quad\mid\quad \mathbb{T}\psi}\;\rightarrow\mathbb{T} \qquad\qquad \frac{\mathbb{F}\varphi\rightarrow\psi}{\begin{array}{c}\mathbb{T}\varphi\\ \mathbb{F}\psi\end{array}}\;\rightarrow\mathbb{F}$$

The Cut Rule

$$\frac{}{\mathbb{T}\varphi \quad\mid\quad \mathbb{F}\varphi}\;\text{Cut}$$

The Cut rule splits every branch in two. It is not necessary (any set of signed formulas with a closed tableau has one not using Cut), but it allows convenient combination of tableaux.

4.5.3 Quantifier Rules

Rules for $\forall$

$$\frac{\mathbb{T}\forall x\,\varphi(x)}{\mathbb{T}\varphi(t)}\;\forall\mathbb{T} \qquad\qquad \frac{\mathbb{F}\forall x\,\varphi(x)}{\mathbb{F}\varphi(a)}\;\forall\mathbb{F}$$

In $\forall\mathbb{T}$, t is a closed term. In $\forall\mathbb{F}$, a is a constant symbol not occurring anywhere in the branch above. We call a the *eigenvariable*.

Rules for $\exists$

$$\frac{\mathbb{T}\,\exists x\,\varphi(x)}{\mathbb{T}\,\varphi(a)}\;\exists\mathbb{T} \qquad\qquad \frac{\mathbb{F}\,\exists x\,\varphi(x)}{\mathbb{F}\,\varphi(t)}\;\exists\mathbb{F}$$

Again, t is a closed term, and a is a constant symbol not occurring in the branch above. The *eigenvariable condition* requires that a not occur in the branch above the $\forall\mathbb{F}$ or $\exists\mathbb{T}$ inference.

Identity Rules

$$\frac{}{\mathbb{T}\,t = t}\;= \qquad \begin{array}{c}\mathbb{T}\,t_1 = t_2\\ \mathbb{T}\,\varphi(t_1)\\ \hline \mathbb{T}\,\varphi(t_2)\end{array}\;=\mathbb{T} \qquad \begin{array}{c}\mathbb{T}\,t_1 = t_2\\ \mathbb{F}\,\varphi(t_1)\\ \hline \mathbb{F}\,\varphi(t_2)\end{array}\;=\mathbb{F}$$

In contrast to the other rules, $=\mathbb{T}$ and $=\mathbb{F}$ require two signed formulas already on the branch: both $\mathbb{T}\,t_1 = t_2$ and $S\,\varphi(t_1)$.

Example 4.66. To show that $p \to p$ is a tautology, we construct a tableau starting from $\mathbb{F}\,p \to p$:

$$\begin{array}{lll} 1. & \mathbb{F}\,p \to p & \\ 2. & \mathbb{T}\,p & (\to\mathbb{F}, 1) \\ 3. & \mathbb{F}\,p & (\to\mathbb{F}, 1) \\ & \otimes & (2, 3) \end{array}$$

The $\to\mathbb{F}$ rule applied to line 1 produces both $\mathbb{T}\,p$ and $\mathbb{F}\,p$ on the same branch. Since p appears with both signs, the branch closes ($\otimes$). Every branch being closed, the tableau is closed, confirming that $p \to p$ is valid.

4.5.4 Tableaux

Definition 4.67 (Tableau)**.** A *tableau* for assumptions $S_1\varphi_1, \ldots, S_n\varphi_n$ (where each S_i is $\mathbb{T}$ or $\mathbb{F}$) is a finite tree of signed formulas satisfying:

1. The n topmost signed formulas are $S_i\varphi_i$, one below the other.
2. Every signed formula not among the assumptions results from a correct application of an inference rule to a signed formula in the branch above it.

A branch is *closed* iff it contains both $\mathbb{T}\,\varphi$ and $\mathbb{F}\,\varphi$, and *open* otherwise. A tableau with every branch closed is a *closed tableau*; otherwise it is *open*.

4.5.5 Soundness

Definition 4.68 (Satisfaction of Signed Formulas)**.** A structure $\mathfrak{M}$ *satisfies* $\mathbb{T}\,\varphi$ iff $\mathfrak{M} \vDash \varphi$, and satisfies $\mathbb{F}\,\varphi$ iff $\mathfrak{M} \nvDash \varphi$. $\mathfrak{M}$ satisfies a set Γ of signed formulas iff it satisfies every member. Γ is *satisfiable* if some structure satisfies it, and *unsatisfiable* otherwise.

Theorem 4.69 (Soundness)**.** *If Γ has a closed tableau, Γ is unsatisfiable.*

Proof. Call a branch *satisfiable* iff the set of signed formulas on it is satisfiable, and call a tableau *satisfiable* if it has at least one satisfiable branch.

We show: extending a satisfiable tableau by one rule of inference always results in a satisfiable tableau. This proves the theorem: a closed tableau results from applying rules to the tableau consisting only of the assumptions from Γ. If Γ were satisfiable, the initial tableau would be satisfiable, and hence every extension would be satisfiable. But a closed tableau is clearly not satisfiable: every branch contains both $\mathbb{T}\,\varphi$ and $\mathbb{F}\,\varphi$.

Suppose we have a satisfiable tableau with a satisfiable branch to which a rule is applied. Let Γ be the set of signed formulas on that branch, and let $S\,\varphi \in \Gamma$ be the signed formula to which the rule is applied. If the rule does not split, we show that the extended branch is satisfiable. If the rule splits, we show at least one resulting branch is satisfiable.

Non-splitting rules:

1. $\neg\mathbb{T}$ *applied to* $\mathbb{T}\neg\psi$*:* The extended branch contains $\Gamma \cup \{\mathbb{F}\,\psi\}$. If $\mathfrak{M} \vDash \Gamma$, then $\mathfrak{M} \vDash \neg\psi$, so $\mathfrak{M} \nvDash \psi$, i.e., $\mathfrak{M}$ satisfies $\mathbb{F}\,\psi$.

2. $\neg\mathbb{F}$ *applied to* $\mathbb{F}\,\neg\psi$*:* If $\mathfrak{M} \nvDash \neg\psi$, then $\mathfrak{M} \vDash \psi$, so $\mathfrak{M}$ satisfies $\mathbb{T}\,\psi$.

3. $\wedge\mathbb{T}$ *applied to* $\mathbb{T}\,\psi \wedge \chi$*:* If $\mathfrak{M} \vDash \psi \wedge \chi$, then $\mathfrak{M} \vDash \psi$ and $\mathfrak{M} \vDash \chi$, so $\mathfrak{M}$ satisfies both new signed formulas.

4. $\vee\mathbb{F}$ *applied to* $\mathbb{F}\,\psi \vee \chi$*:* If $\mathfrak{M} \nvDash \psi \vee \chi$, then $\mathfrak{M} \nvDash \psi$ and $\mathfrak{M} \nvDash \chi$.

5. $\rightarrow\mathbb{F}$ *applied to* $\mathbb{F}\,\psi \rightarrow \chi$*:* If $\mathfrak{M} \nvDash \psi \rightarrow \chi$, then $\mathfrak{M} \vDash \psi$ and $\mathfrak{M} \nvDash \chi$.

6. $\forall\mathbb{T}$ *applied to* $\mathbb{T}\forall x\,\varphi(x)$*:* This adds $\mathbb{T}\,\varphi(t)$. If $\mathfrak{M} \vDash \forall x\,\varphi(x)$, by the substitution lemma $\mathfrak{M} \vDash \varphi(t)$.

7. $\forall\mathbb{F}$ *applied to* $\mathbb{F}\forall x\,\varphi(x)$*:* This adds $\mathbb{F}\,\varphi(a)$ where a does not occur in Γ. Since $\mathfrak{M} \nvDash \forall x\,\varphi(x)$, for some variable assignment s, $\mathfrak{M}, s \nvDash \varphi(x)$. Let $\mathfrak{M}'$ be like $\mathfrak{M}$ except $a^{\mathfrak{M}'} = s(x)$. Since a does not occur in Γ, $\mathfrak{M}'$ still satisfies Γ. By the substitution lemma, $\mathfrak{M}' \nvDash \varphi(a)$, so $\mathfrak{M}'$ satisfies $\mathbb{F}\,\varphi(a)$.

8. $\exists\mathbb{T}$ *applied to* $\mathbb{T}\exists x\,\varphi(x)$*:* Symmetric to the $\forall\mathbb{F}$ case, constructing $\mathfrak{M}'$ with $a^{\mathfrak{M}'} = s(x)$ for a suitable s.

9. $\exists\mathbb{F}$ *applied to* $\mathbb{F}\exists x\,\varphi(x)$*:* Symmetric to the $\forall\mathbb{T}$ case.

10. *Identity rule* $=$ (adding $\mathbb{T}t = t$): Trivially $\mathfrak{M} \vDash t = t$.

11. $=\mathbb{T}$ (adding $\mathbb{T}\varphi(t_2)$ from $\mathbb{T}t_1 = t_2$ and $\mathbb{T}\varphi(t_1)$): Since $\mathrm{Val}^{\mathfrak{M}}(t_1) = \mathrm{Val}^{\mathfrak{M}}(t_2)$, the substitution lemma gives $\mathfrak{M} \vDash \varphi(t_2)$. The $=\mathbb{F}$ case is similar.

Splitting rules:

1. $\wedge\mathbb{F}$ *applied to* $\mathbb{F}\psi \wedge \chi$*:* Splits into $\mathbb{F}\psi$ and $\mathbb{F}\chi$. If $\mathfrak{M} \nvDash \psi \wedge \chi$, then $\mathfrak{M} \nvDash \psi$ or $\mathfrak{M} \nvDash \chi$; accordingly $\mathfrak{M}$ satisfies the left or right branch.

2. $\vee\mathbb{T}$ *applied to* $\mathbb{T}\psi \vee \chi$*:* Splits into $\mathbb{T}\psi$ and $\mathbb{T}\chi$. Since $\mathfrak{M} \vDash \psi$ or $\mathfrak{M} \vDash \chi$, at least one branch is satisfiable.

3. $\to\mathbb{T}$ *applied to* $\mathbb{T}\psi \to \chi$*:* Splits into $\mathbb{F}\psi$ and $\mathbb{T}\chi$. Since either $\mathfrak{M} \nvDash \psi$ or $\mathfrak{M} \vDash \chi$, at least one branch is satisfiable.

4. Cut*:* Splits into $\mathbb{T}\psi$ and $\mathbb{F}\psi$. Since either $\mathfrak{M} \vDash \psi$ or $\mathfrak{M} \nvDash \psi$, at least one branch is satisfiable.

□

Corollary 4.70 (Weak Soundness)**.** *If* $\vdash \varphi$ *then* φ *is valid.*

Corollary 4.71. *If* $\Gamma \vdash \varphi$ *then* $\Gamma \vDash \varphi$*.*

Proof. If $\Gamma \vdash \varphi$ then for some $\psi_1, \ldots, \psi_n \in \Gamma$, $\{\mathbb{F}\varphi, \mathbb{T}\psi_1, \ldots, \mathbb{T}\psi_n\}$ has a closed tableau. By Theorem 4.69, every structure $\mathfrak{M}$ either falsifies some ψ_i or satisfies φ. Hence if $\mathfrak{M} \vDash \Gamma$ then $\mathfrak{M} \vDash \varphi$. □

Corollary 4.72. *If* Γ *is satisfiable, then it is consistent.*

Proof. Contrapositive: if Γ is inconsistent, then there are $\psi_1, \ldots, \psi_n \in \Gamma$ with a closed tableau for $\{\mathbb{T}\psi_1, \ldots, \mathbb{T}\psi_n\}$. By Theorem 4.69, no structure satisfies all ψ_i, so Γ is unsatisfiable. □

Remark 22. Tableau consistency—the absence of a closed tableau from $\mathbb{T}$-signed assumptions—is a reformulation of the generic consistency notion of §4.1. Transitivity uses the Cut rule.

4.6 Theories and Arithmetic

The proof systems of DED.2–DED.5 provide the deductive machinery; this section introduces the formal theories of arithmetic that serve as the principal objects of study in the incompleteness theorems (see CH-BST, Boundedness). We define Robinson's **Q**, Peano Arithmetic **PA**, the notion of ω-consistency, and the derivability conditions that any sufficiently strong theory must satisfy for the incompleteness theorems to apply.

4.6.1 Robinson Arithmetic Q

The natural language in which to express facts of arithmetic is $\mathcal{L}_A$. $\mathcal{L}_A$ contains a single two-place predicate symbol $<$, a single constant symbol o, one one-place function symbol $\prime$, and two two-place function symbols $+$ and $\times$ (see PRIM-SYN009, Language, §2.1 for the general notion of a first-order language).

Definition 4.73 (Robinson Arithmetic **Q**)**.** The theory **Q** axiomatized by the following sentences is known as "Robinson's **Q**" and is a very simple theory of arithmetic.

$$\forall x\, \forall y\, (x' = y' \to x = y) \tag{Q_1}$$
$$\forall x\, \mathsf{o} \neq x' \tag{Q_2}$$
$$\forall x\, (x = \mathsf{o} \lor \exists y\, x = y') \tag{Q_3}$$
$$\forall x\, (x + \mathsf{o}) = x \tag{Q_4}$$
$$\forall x\, \forall y\, (x + y') = (x + y)' \tag{Q_5}$$
$$\forall x\, (x \times \mathsf{o}) = \mathsf{o} \tag{Q_6}$$
$$\forall x\, \forall y\, (x \times y') = ((x \times y) + x) \tag{Q_7}$$
$$\forall x\, \forall y\, (x < y \leftrightarrow \exists z\, (z' + x) = y) \tag{Q_8}$$

The sentences $\{Q_1, \dots, Q_8\}$ are the axioms of **Q**, so

$$\mathbf{Q} = \{\varphi : \{Q_1, \dots, Q_8\} \vDash \varphi\}.$$

The axioms Q_1 and Q_2 express that the successor function is injective with 0 not in its range. Axiom Q_3 says every number is either 0 or a successor. Axioms Q_4–Q_7 give the recursive definitions of addition and multiplication. Axiom Q_8 defines the ordering in terms of addition.

Q is weak: it cannot even prove the commutativity of addition. Its importance lies in the fact that **Q** is strong enough to represent all computable functions and all decidable relations (see §5.4), which is the key hypothesis of the incompleteness theorems. Since any theory that extends **Q** inherits this representability, the incompleteness theorems apply to a wide class of theories.

4.6.2 Peano Arithmetic PA

Definition 4.74 (Peano Arithmetic **PA**)**.** Suppose $\varphi(x)$ is a formula in $\mathcal{L}_A$ with free variables x and $y_1, \ldots, y_n$. Then any sentence of the form

$$\forall y_1 \ldots \forall y_n \left((\varphi(\mathrm{o}) \land \forall x\, (\varphi(x) \to \varphi(x'))) \to \forall x\, \varphi(x) \right)$$

is an instance of the *induction schema*.

Peano Arithmetic **PA** is the theory axiomatized by the axioms of **Q** together with all instances of the induction schema.

Every instance of the induction schema is true in the standard model of arithmetic $\mathfrak{N}$ (see DEF-SEM012, Standard Model, §3.5). If $\varphi(x)$ defines a subset X_φ of $\mathbb{N}$ in $\mathfrak{N}$, then the induction schema asserts that if $0 \in X_\varphi$ and X_φ is closed under the successor function, then $X_\varphi = \mathbb{N}$.

The induction schema is genuinely a *schema*: it generates infinitely many axioms, and **PA** is not finitely axiomatizable. However, since one can effectively determine whether a string of symbols is an instance of an induction axiom, the set of axioms for **PA** is decidable, and **PA** is an axiomatizable theory in the sense of PRIM-DED002.

PA is a much more robust theory than **Q**: one can prove the commutativity and associativity of addition and multiplication, and in fact most finitary number-theoretic and combinatorial arguments can be carried out in **PA**.

4.6.3 ω-Consistency

Definition 4.75 (ω-Consistency)**.** A theory **T** is *ω-consistent* if the following holds: if $\exists x\, \varphi(x)$ is any sentence and **T** proves $\neg\varphi(\overline{0})$, $\neg\varphi(\overline{1})$, $\neg\varphi(\overline{2})$, …, then **T** does not prove $\exists x\, \varphi(x)$.

ω-consistency is strictly stronger than ordinary consistency: every ω-consistent theory is consistent, but the converse fails. Gödel's original 1931 proof of the first incompleteness theorem assumed ω-consistency. Rosser subsequently strengthened the result by replacing ω-consistency with ordinary consistency (see Rosser's Theorem, §6.5).

4.6.4 Derivability Conditions

Definition 4.76 (Derivability Conditions)**.** Let **T** be an axiomatizable theory extending **Q**, and let $\mathsf{Prov}_{\mathbf{T}}(y)$ be a formula representing the derivability predicate for **T** (i.e., $\mathsf{Prov}_{\mathbf{T}}(y) = \exists x\, \mathsf{Prf}_{\mathbf{T}}(x, y)$, where $\mathsf{Prf}_{\mathbf{T}}(x, y)$ represents the proof relation). The *derivability conditions* (also called Hilbert–Bernays–Löb conditions) are:

P1. If $\mathbf{T} \vdash \varphi$, then $\mathbf{T} \vdash \mathsf{Prov}_{\mathbf{T}}(\ulcorner \varphi \urcorner)$.

P2. For all formulas φ and ψ,

$$\mathbf{T} \vdash \mathsf{Prov}_{\mathbf{T}}(\ulcorner \varphi \to \psi \urcorner) \to (\mathsf{Prov}_{\mathbf{T}}(\ulcorner \varphi \urcorner) \to \mathsf{Prov}_{\mathbf{T}}(\ulcorner \psi \urcorner)).$$

P3. For every formula φ,

$$\mathbf{T} \vdash \mathsf{Prov}_{\mathbf{T}}(\ulcorner \varphi \urcorner) \to \mathsf{Prov}_{\mathbf{T}}(\ulcorner \mathsf{Prov}_{\mathbf{T}}(\ulcorner \varphi \urcorner) \urcorner).$$

Condition P1 says that **T** is aware of its own theorems: if it proves φ, it proves that it proves φ. Condition P2 says that the provability predicate distributes over the conditional, so that **T** can reason internally about modus ponens. Condition P3 is a form of positive introspection: **T** can verify its own provability.

All three conditions hold for **PA** (and, more generally, for any axiomatizable extension of **Q** with a suitably chosen provability predicate). Conditions P1 and P2 are relatively easy to verify; P3 requires substantial formalization of proof theory inside **T** itself.

The derivability conditions are the essential hypotheses for Löb's Theorem (see THM-DED007, §4.7) and the Second Incompleteness Theorem (see CP-006, §6.6).

4.7 Theorems

The proof systems and formal theories of the preceding sections support several deep results that bridge deduction and metatheory.

4.7.1 The Deduction Theorem

Theorem 4.77 (Deduction Theorem)**.** *$\Gamma \cup \{\varphi\} \vdash \psi$ if and only if $\Gamma \vdash \varphi \to \psi$.*

The deduction theorem is a fundamental metatheorem relating assumption introduction to the conditional connective. Its proof depends on the proof system:

- In axiomatic deduction, the proof is by induction on the length of the derivation and uses the logical axiom $\varphi \to (\psi \to \varphi)$ and the distribution axiom $(\varphi \to (\psi \to \chi)) \to ((\varphi \to \psi) \to (\varphi \to \chi))$. See DED.2 for the full proof.

- In natural deduction, the deduction theorem is immediate from the $\to$Intro rule, which allows one to discharge an assumption φ and conclude $\varphi \to \psi$. See DED.3.
- In the sequent calculus, the deduction theorem corresponds to the right conditional rule $\to$R, which moves a formula from the antecedent to the succedent. See DED.4.

4.7.2 Lindenbaum's Lemma

Lemma 4.78 (Lindenbaum's Lemma)**.** *Every consistent set Γ in a language $\mathcal{L}$ can be extended to a complete and consistent set Γ^*.*

Proof. Let Γ be consistent. Let $\varphi_0, \varphi_1, \ldots$ be an enumeration of all the sentences of $\mathcal{L}$. Define $\Gamma_0 = \Gamma$, and

$$\Gamma_{n+1} = \begin{cases} \Gamma_n \cup \{\varphi_n\} & \text{if } \Gamma_n \cup \{\varphi_n\} \text{ is consistent;} \\ \Gamma_n \cup \{\neg\varphi_n\} & \text{otherwise.} \end{cases}$$

Let $\Gamma^* = \bigcup_{n \geq 0} \Gamma_n$.

Each Γ_n is consistent: Γ_0 is consistent by definition. If $\Gamma_{n+1} = \Gamma_n \cup \{\varphi_n\}$, this is because the latter is consistent. If it is not, $\Gamma_{n+1} = \Gamma_n \cup \{\neg\varphi_n\}$. We verify that $\Gamma_n \cup \{\neg\varphi_n\}$ is consistent. If it were not, then *both* $\Gamma_n \cup \{\varphi_n\}$ and $\Gamma_n \cup \{\neg\varphi_n\}$ would be inconsistent. By Proposition 4.20, Γ_n would be inconsistent, contradicting the induction hypothesis.

For every n and every $i < n$, $\Gamma_i \subseteq \Gamma_n$. This follows by a simple induction on n.

From this it follows that Γ^* is consistent: let $\Gamma' \subseteq \Gamma^*$ be finite. Each $\psi \in \Gamma'$ is also in Γ_i for some i. Let n be the largest of these. Since $\Gamma_i \subseteq \Gamma_n$ if $i \leq n$, every $\psi \in \Gamma'$ is also $\in \Gamma_n$, i.e., $\Gamma' \subseteq \Gamma_n$, and Γ_n is consistent. So every finite subset $\Gamma' \subseteq \Gamma^*$ is consistent. By compactness (Proposition 4.15), Γ^* is consistent.

Every sentence of $\mathrm{Frm}(\mathcal{L})$ appears on the list used to define Γ^*. If $\varphi_n \notin \Gamma^*$, then that is because $\Gamma_n \cup \{\varphi_n\}$ was inconsistent. But then $\neg\varphi_n \in \Gamma^*$, so Γ^* is complete. □

4.7.3 The Fixed-Point Lemma

The fixed-point lemma (also called the diagonal lemma or self-referential lemma) is the engine behind the incompleteness theorems. It guarantees that any expressible property can be asserted of its own Gödel number.

Lemma 4.79 (Fixed-Point Lemma). *Let $\psi(x)$ be any formula with one free variable x. Then there is a sentence φ such that $\mathbf{Q} \vdash \varphi \leftrightarrow \psi(\ulcorner\varphi\urcorner)$.*

Proof. Given $\psi(x)$, let $\alpha(x)$ be the formula $\exists y\,(\theta_{\text{diag}}(x, y) \land \psi(y))$, where $\theta_{\text{diag}}(x, y)$ is a formula representing the primitive recursive diagonalization function diag in $\mathbf{Q}$. The function $\text{diag}(n)$ computes, given the Gödel number n of a formula $\alpha(x)$, the Gödel number of its diagonalization $\alpha(\ulcorner\alpha(x)\urcorner)$.

Let φ be the diagonalization of $\alpha(x)$, i.e., φ is $\alpha(\ulcorner\alpha(x)\urcorner)$.

Since θ_{diag} represents diag, and $\text{diag}(^{\#}\alpha(x)^{\#}) = {}^{\#}\varphi^{\#}$, $\mathbf{Q}$ can derive:

$$\theta_{\text{diag}}(\ulcorner\alpha(x)\urcorner, \ulcorner\varphi\urcorner) \tag{4.1}$$

$$\forall y\,(\theta_{\text{diag}}(\ulcorner\alpha(x)\urcorner, y) \to y = \ulcorner\varphi\urcorner). \tag{4.2}$$

We show that $\mathbf{Q} \vdash \varphi \leftrightarrow \psi(\ulcorner\varphi\urcorner)$, arguing informally using logic and facts derivable in $\mathbf{Q}$.

Forward direction. Suppose φ, i.e., $\alpha(\ulcorner\alpha(x)\urcorner)$, which by definition of $\alpha(x)$ is

$$\exists y\,(\theta_{\text{diag}}(\ulcorner\alpha(x)\urcorner, y) \land \psi(y)).$$

Consider such a y. Since $\theta_{\text{diag}}(\ulcorner\alpha(x)\urcorner, y)$, by (6.2), $y = \ulcorner\varphi\urcorner$. So from $\psi(y)$ we have $\psi(\ulcorner\varphi\urcorner)$.

Reverse direction. Suppose $\psi(\ulcorner\varphi\urcorner)$. By (6.1), we have

$$\theta_{\text{diag}}(\ulcorner\alpha(x)\urcorner, \ulcorner\varphi\urcorner) \land \psi(\ulcorner\varphi\urcorner).$$

It follows that

$$\exists y\,(\theta_{\text{diag}}(\ulcorner\alpha(x)\urcorner, y) \land \psi(y)),$$

which is $\alpha(\ulcorner\alpha(x)\urcorner)$, i.e., φ. □

4.7.4 Löb's Theorem

Theorem 4.80 (Löb's Theorem). *Let $\mathbf{T}$ be an axiomatizable theory extending $\mathbf{Q}$, and suppose $\mathsf{Prov}_{\mathbf{T}}(y)$ is a formula satisfying the derivability conditions P1–P3 (see DEF-DED014, §4.6). If $\mathbf{T}$ derives $\mathsf{Prov}_{\mathbf{T}}(\ulcorner\varphi\urcorner) \to \varphi$, then in fact $\mathbf{T}$ derives φ.*

Equivalently: if $\mathbf{T} \nvdash \varphi$, then $\mathbf{T} \nvdash \mathsf{Prov}_{\mathbf{T}}(\ulcorner\varphi\urcorner) \to \varphi$. The schema $\mathsf{Prov}_{\mathbf{T}}(\ulcorner\varphi\urcorner) \to \varphi$ is called the *reflection principle*; Löb's theorem says that a consistent theory can only derive those instances of the reflection principle where φ is already a theorem.

Proof idea. Apply the fixed-point lemma to the formula $\mathsf{Prov}_{\mathbf{T}}(y) \to \varphi$, obtaining a sentence θ with $\mathbf{T} \vdash \theta \leftrightarrow (\mathsf{Prov}_{\mathbf{T}}(\ulcorner\theta\urcorner) \to \varphi)$. Informally, θ says "if I am provable, then φ." The derivability conditions P1–P3 then show $\mathbf{T} \vdash \theta$, whence $\mathbf{T} \vdash \varphi$.

Proof. Suppose φ is a sentence such that **T** derives $\mathsf{Prov}_{\mathbf{T}}(\ulcorner\varphi\urcorner) \to \varphi$. Let $\psi(y)$ be the formula $\mathsf{Prov}_{\mathbf{T}}(y) \to \varphi$, and use the fixed-point lemma (THM-DED006) to find a sentence θ such that **T** derives $\theta \leftrightarrow \psi(\ulcorner\theta\urcorner)$. Then each of the following is derivable in **T**:

$$\theta \leftrightarrow (\mathsf{Prov}_{\mathbf{T}}(\ulcorner\theta\urcorner) \to \varphi) \tag{4.3}$$

θ is a fixed point of $\psi(y)$

$$\theta \to (\mathsf{Prov}_{\mathbf{T}}(\ulcorner\theta\urcorner) \to \varphi) \tag{4.4}$$

from (6.14)

$$\mathsf{Prov}_{\mathbf{T}}(\ulcorner\theta \to (\mathsf{Prov}_{\mathbf{T}}(\ulcorner\theta\urcorner) \to \varphi)\urcorner) \tag{4.5}$$

from (6.15) by condition P1

$$\mathsf{Prov}_{\mathbf{T}}(\ulcorner\theta\urcorner) \to \mathsf{Prov}_{\mathbf{T}}(\ulcorner\mathsf{Prov}_{\mathbf{T}}(\ulcorner\theta\urcorner) \to \varphi\urcorner) \tag{4.6}$$

from (6.16) using condition P2

$$\mathsf{Prov}_{\mathbf{T}}(\ulcorner\theta\urcorner) \to (\mathsf{Prov}_{\mathbf{T}}(\ulcorner\mathsf{Prov}_{\mathbf{T}}(\ulcorner\theta\urcorner)\urcorner) \to \mathsf{Prov}_{\mathbf{T}}(\ulcorner\varphi\urcorner)) \tag{4.7}$$

from (6.17) using P2 again

$$\mathsf{Prov}_{\mathbf{T}}(\ulcorner\theta\urcorner) \to \mathsf{Prov}_{\mathbf{T}}(\ulcorner\mathsf{Prov}_{\mathbf{T}}(\ulcorner\theta\urcorner)\urcorner) \tag{4.8}$$

by derivability condition P3

$$\mathsf{Prov}_{\mathbf{T}}(\ulcorner\theta\urcorner) \to \mathsf{Prov}_{\mathbf{T}}(\ulcorner\varphi\urcorner) \tag{4.9}$$

from (6.18) and (6.19)

$$\mathsf{Prov}_{\mathbf{T}}(\ulcorner\varphi\urcorner) \to \varphi \tag{4.10}$$

by assumption of the theorem

$$\mathsf{Prov}_{\mathbf{T}}(\ulcorner\theta\urcorner) \to \varphi \tag{4.11}$$

from (6.20) and (6.21)

$$(\mathsf{Prov}_{\mathbf{T}}(\ulcorner\theta\urcorner) \to \varphi) \to \theta \tag{4.12}$$

from (6.14)

$$\theta \tag{4.13}$$

from (6.22) and (6.23)

$$\mathsf{Prov}_{\mathbf{T}}(\ulcorner\theta\urcorner) \tag{4.14}$$

from (6.24) by condition P1

φ from (6.21) and (6.25)

□

Remark 23. With Löb's theorem in hand, there is a short proof of the second incompleteness theorem (see CP-006, Second Incompleteness Theorem, §6.6). Take $\varphi = \bot$. If $\mathbf{T} \vdash \mathsf{Prov}_{\mathbf{T}}(\ulcorner\bot\urcorner) \to \bot$, then by Löb's theorem $\mathbf{T} \vdash \bot$. Contrapositively, if **T** is consistent, then $\mathbf{T} \nvdash \mathsf{Prov}_{\mathbf{T}}(\ulcorner\bot\urcorner) \to \bot$, i.e., $\mathbf{T} \nvdash \mathsf{Con}_{\mathbf{T}}$.

Löb's theorem also settles the status of the fixed point δ of $\mathsf{Prov}_{\mathbf{T}}(x)$, i.e., a sentence δ such that $\mathbf{T} \vdash \mathsf{Prov}_{\mathbf{T}}(\ulcorner\delta\urcorner) \leftrightarrow \delta$. Since in particular $\mathbf{T} \vdash \mathsf{Prov}_{\mathbf{T}}(\ulcorner\delta\urcorner) \rightarrow \delta$, Löb's theorem gives $\mathbf{T} \vdash \delta$.

Chapter 5

Computation

5.1 Recursive Functions

We now develop the theory of computable functions on the natural numbers, beginning with the primitive recursive functions and extending to the partial recursive functions via unbounded search. The primitive recursive functions form a robust class closed under composition and primitive recursion, but they do not exhaust the computable functions. To capture computability in full, we must allow partial functions and the μ-operator.

5.1.1 Partial and total functions

We work throughout with functions whose domain and codomain are subsets of $\mathbb{N}$. A function may fail to be defined on some inputs; we make this precise.

Definition 5.1 (Partial function)**.** A *partial function* $f\colon \mathbb{N}^k \nrightarrow \mathbb{N}$ is a function from a subset of $\mathbb{N}^k$ to $\mathbb{N}$. We write $f(\vec{x})\downarrow$ to mean that f is defined at $\vec{x}$ (i.e., $\vec{x}$ is in the domain of f), and $f(\vec{x})\uparrow$ to mean that f is not defined at $\vec{x}$. We write $f(\vec{x}) \simeq g(\vec{x})$ to mean that either both $f(\vec{x})$ and $g(\vec{x})$ are undefined, or both are defined and equal.

Definition 5.2 (Total function)**.** A partial function $f\colon \mathbb{N}^k \nrightarrow \mathbb{N}$ is *total* if $f(\vec{x})\downarrow$ for every $\vec{x} \in \mathbb{N}^k$, i.e., if its domain is all of $\mathbb{N}^k$.

We adopt the convention that if h and $g_0, \dots, g_k$ are all partial functions, then $h(g_0(\vec{x}), \dots, g_k(\vec{x}))$ is defined if and only if each g_i is defined at $\vec{x}$, and h is defined at $g_0(\vec{x}), \dots, g_k(\vec{x})$.

5.1.2 Composition and primitive recursion

Definition 5.3 (Composition)**.** Suppose f is a k-place function, and g_0, …, g_{k-1} are k functions which are all n-place. The function defined by *composition from f and $g_0, \dots, g_{k-1}$* is the n-place function h defined by

$$h(x_0, \dots, x_{n-1}) = f(g_0(x_0, \dots, x_{n-1}), \dots, g_{k-1}(x_0, \dots, x_{n-1})).$$

Together with the projection functions $P^n_i(x_0, \dots, x_{n-1}) = x_i$, composition provides sufficient flexibility to rearrange, duplicate, or discard arguments. For instance, a three-place function $g(x_0, y, z) = \mathrm{succ}(z)$ can be defined as $g(x_0, y, z) = \mathrm{succ}(P^3_2(x_0, y, z))$.

Definition 5.4 (Primitive recursion)**.** Suppose f is a k-place function ($k \geq 1$) and g is a $(k+2)$-place function. The function defined by *primitive recursion from f and g* is the $(k+1)$-place function h defined by the equations

$$\begin{aligned} h(x_0, \dots, x_{k-1}, 0) &= f(x_0, \dots, x_{k-1}) \\ h(x_0, \dots, x_{k-1}, y+1) &= g(x_0, \dots, x_{k-1}, y, h(x_0, \dots, x_{k-1}, y)) \end{aligned}$$

Definition 5.5 (Primitive recursive function)**.** The set of *primitive recursive functions* is the set of functions from $\mathbb{N}^n$ to $\mathbb{N}$, defined inductively by the following clauses:

1. zero is primitive recursive.
2. succ is primitive recursive.
3. Each projection function P^n_i is primitive recursive.
4. If f is a k-place primitive recursive function and g_0, …, g_{k-1} are n-place primitive recursive functions, then the composition of f with g_0, …, g_{k-1} is primitive recursive.
5. If f is a k-place primitive recursive function and g is a $k+2$-place primitive recursive function, then the function defined by primitive recursion from f and g is primitive recursive.

Equivalently, the set of primitive recursive functions is the smallest set containing zero, succ, and the projection functions P^n_j, and which is closed under composition and primitive recursion.

5.1.3 Primitive recursive relations

Definition 5.6 (Characteristic function)**.** The *characteristic function* of a relation $R(\vec{x})$ is the function

$$\chi_R(\vec{x}) = \begin{cases} 1 & \text{if } R(\vec{x}) \\ 0 & \text{otherwise.} \end{cases}$$

A relation $R(\vec{x})$ is said to be *primitive recursive* if its characteristic function χ_R is primitive recursive.

For example, the relation $\mathrm{IsZero}(x)$, which holds if and only if $x = 0$, corresponds to the function χ_{IsZero} defined by primitive recursion: $\chi_{\mathrm{IsZero}}(0) = 1$ and $\chi_{\mathrm{IsZero}}(x+1) = 0$. The equality relation $x = y$ is primitive recursive, defined by $\mathrm{IsZero}(|x-y|)$, and the ordering $x \leq y$ is primitive recursive, defined by $\mathrm{IsZero}(x \dot{-} y)$.

Proposition 5.7. *The set of primitive recursive relations is closed under Boolean operations, that is, if $P(\vec{x})$ and $Q(\vec{x})$ are primitive recursive, so are*

1. $\neg P(\vec{x})$
2. $P(\vec{x}) \land Q(\vec{x})$
3. $P(\vec{x}) \lor Q(\vec{x})$
4. $P(\vec{x}) \to Q(\vec{x})$

Proof. Suppose $P(\vec{x})$ and $Q(\vec{x})$ are primitive recursive, i.e., their characteristic functions χ_P and χ_Q are. We have to show that the characteristic functions of $\neg P(\vec{x})$, etc., are also primitive recursive.

$$\chi_{\neg P}(\vec{x}) = \begin{cases} 0 & \text{if } \chi_P(\vec{x}) = 1 \\ 1 & \text{otherwise} \end{cases}$$

We can define $\chi_{\neg P}(\vec{x})$ as $1 \dot{-} \chi_P(\vec{x})$.

$$\chi_{P \land Q}(\vec{x}) = \begin{cases} 1 & \text{if } \chi_P(\vec{x}) = \chi_Q(\vec{x}) = 1 \\ 0 & \text{otherwise} \end{cases}$$

We can define $\chi_{P \land Q}(\vec{x})$ as $\chi_P(\vec{x}) \cdot \chi_Q(\vec{x})$ or as $\min(\chi_P(\vec{x}), \chi_Q(\vec{x}))$. Similarly,

$$\chi_{P \lor Q}(\vec{x}) = \max(\chi_P(\vec{x}), \chi_Q(\vec{x})) \text{ and}$$
$$\chi_{P \to Q}(\vec{x}) = \max(1 \dot{-} \chi_P(\vec{x}), \chi_Q(\vec{x})).$$

□

Proposition 5.8. *The set of primitive recursive relations is closed under bounded quantification, i.e., if $R(\vec{x}, z)$ is a primitive recursive relation, then so are the relations*

$$\begin{gathered}(\forall z < y)\ R(\vec{x}, z) \text{ and} \\ (\exists z < y)\ R(\vec{x}, z).\end{gathered}$$

$(\forall z < y)\ R(\vec{x}, z)$ holds of $\vec{x}$ and y if and only if $R(\vec{x}, z)$ holds for every z less than y, and similarly for $(\exists z < y)\ R(\vec{x}, z)$.

Proof. By convention, we take $(\forall z < 0)\ R(\vec{x}, z)$ to be true (for the trivial reason that there are no z less than 0) and $(\exists z < 0)\ R(\vec{x}, z)$ to be false. A bounded universal quantifier functions like an iterated minimum: if $P(\vec{x}, y) \Leftrightarrow (\forall z < y)\ R(\vec{x}, z)$ then $\chi_P(\vec{x}, y)$ can be defined by

$$\begin{aligned}\chi_P(\vec{x}, 0) &= 1 \\ \chi_P(\vec{x}, y + 1) &= \min(\chi_P(\vec{x}, y), \chi_R(\vec{x}, y)).\end{aligned}$$

Bounded existential quantification can similarly be defined using max. Alternatively, it can be defined from bounded universal quantification, using the equivalence $(\exists z < y)\ R(\vec{x}, z) \leftrightarrow \neg(\forall z < y)\ \neg R(\vec{x}, z)$. Note that a bounded quantifier of the form $(\exists x \leq y)\ \ldots x \ldots$ is equivalent to $(\exists x < y + 1)\ \ldots x \ldots$. □

Proposition 5.9 (Definition by cases)**.** *If $g_0(\vec{x})$, ..., $g_m(\vec{x})$ are primitive recursive functions, and $R_0(\vec{x})$, ..., $R_{m-1}(\vec{x})$ are primitive recursive relations, then the function f defined by*

$$f(\vec{x}) = \begin{cases} g_0(\vec{x}) & \textit{if } R_0(\vec{x}) \\ g_1(\vec{x}) & \textit{if } R_1(\vec{x}) \textit{ and not } R_0(\vec{x}) \\ \vdots & \\ g_{m-1}(\vec{x}) & \textit{if } R_{m-1}(\vec{x}) \textit{ and none of the previous hold} \\ g_m(\vec{x}) & \textit{otherwise} \end{cases}$$

is also primitive recursive.

Proof. The conditional function $\text{cond}(x, y, z)$, defined by $\text{cond}(0, y, z) = y$ and $\text{cond}(x + 1, y, z) = z$, is primitive recursive. When $m = 1$, the function f is just $f(\vec{x}) = \text{cond}(\chi_{\neg R_0}(\vec{x}), g_0(\vec{x}), g_1(\vec{x}))$. For m greater than 1, one composes definitions of this form. □

5.1.4 Bounded minimization

Proposition 5.10. *If $R(\vec{x}, z)$ is primitive recursive, so is the function $m_R(\vec{x}, y)$ which returns the least z less than y such that $R(\vec{x}, z)$ holds, if there is one, and y otherwise. We write this function as*

$$(\min z < y)\, R(\vec{x}, z).$$

Proof. Since there is no $z < 0$, we have $m_R(\vec{x}, 0) = 0$. For the successor case, there are three possibilities: (1) there is a $z < y$ such that $R(\vec{x}, z)$, so $m_R(\vec{x}, y + 1) = m_R(\vec{x}, y)$; (2) there is no such $z < y$ but $R(\vec{x}, y)$ holds, so $m_R(\vec{x}, y + 1) = y$; (3) there is no $z < y + 1$ such that $R(\vec{x}, z)$, so $m_R(\vec{x}, y + 1) = y + 1$. Thus:

$$m_R(\vec{x}, 0) = 0$$

$$m_R(\vec{x}, y+1) = \begin{cases} m_R(\vec{x}, y) & \text{if } m_R(\vec{x}, y) \neq y \\ y & \text{if } m_R(\vec{x}, y) = y \text{ and } R(\vec{x}, y) \\ y+1 & \text{otherwise.} \end{cases}$$

This is a definition by primitive recursion combined with definition by cases from primitive recursive relations, so m_R is primitive recursive. □

Bounded minimization finds the least witness below a given bound. In contrast, the unbounded search operator (introduced next) searches without any bound and may therefore fail to terminate, taking us out of the realm of primitive recursive functions.

5.1.5 Computability of primitive recursive functions

Proposition 5.11. *Every primitive recursive function is computable.*

Proof sketch. The basic functions zero, succ, and P^n_i are computable. Composition preserves computability: to compute $h(\vec{x}) = f(g_0(\vec{x}), \ldots, g_{k-1}(\vec{x}))$, first compute each $g_i(\vec{x})$ and then apply f. Primitive recursion preserves computability: to compute $h(\vec{x}, y)$, successively compute $h(\vec{x}, 0)$, $h(\vec{x}, 1)$, …, until reaching $h(\vec{x}, y)$. Since each step is computable, so is the result. □

5.1.6 Partial recursive functions and unbounded search

We now extend the primitive recursive functions by allowing partial functions and adding the unbounded search operator.

Definition 5.12 (Unbounded search / μ-recursion)**.** If $f(x, \vec{z})$ is any partial function on the natural numbers, define $\mu x\; f(x, \vec{z})$ to be the least x such that $f(0,\vec{z}), f(1,\vec{z}), \dots, f(x,\vec{z})$ are all defined, and $f(x,\vec{z}) = 0$, if such an x exists,

with the understanding that $\mu x\ f(x,\vec{z})$ is undefined otherwise.

If $R(x,\vec{z})$ is any relation, $\mu x\ R(x,\vec{z})$ is defined to be $\mu x\ (1 \dot{-} \chi_R(x,\vec{z}))$, i.e., the least x such that $R(x,\vec{z})$ holds.

Computationally, the procedure for computing $\mu x\ f(x,\vec{z})$ amounts to computing $f(0,\vec{z}), f(1,\vec{z}), f(2,\vec{z}), \dots$ until a value of 0 is returned. If any intermediate computation does not halt, neither does the computation of $\mu x\ f(x,\vec{z})$.

Definition 5.13 (Partial recursive function)**.** The set of *partial recursive functions* is the smallest set of partial functions from the natural numbers to the natural numbers (of various arities) containing zero, successor, and projections, and closed under composition, primitive recursion, and unbounded search.

Definition 5.14 (Recursive function)**.** The set of *recursive functions* (also called *total recursive functions*) is the set of partial recursive functions that are total.

5.2 Turing Machines

We now introduce Turing machines, an independent model of computation that approaches computability from a concrete, mechanical perspective rather than the function-algebraic perspective of the recursive functions.

5.2.1 Definition of Turing machines

Definition 5.15 (Turing machine)**.** A *Turing machine* M is a tuple $\langle Q, \Sigma, q_0, \delta \rangle$ consisting of

1. a finite set of *states* Q,
2. a finite *alphabet* Σ which includes $\rhd$ and 0,
3. an *initial state* $q_0 \in Q$,
4. a finite *instruction set* $\delta \colon Q \times \Sigma \nrightarrow Q \times \Sigma \times \{L, R, N\}$.

The partial function δ is also called the *transition function* of M.

We assume the tape is infinite in one direction only. The symbol $\rhd$ serves as a marker for the left end of the tape, making it possible for programs to detect when they are at the leftmost square.

5.2.2 Configurations and computations

Definition 5.16 (Configuration). A *configuration* of Turing machine $M = \langle Q, \Sigma, q_0, \delta \rangle$ is a triple $\langle C, m, q \rangle$ where

1. $C \in \Sigma^*$ is a finite sequence of symbols from Σ,
2. $m \in \mathbb{N}$ is a number $< \mathrm{len}(C)$, and
3. $q \in Q$.

Intuitively, the sequence C is the content of the tape (from the leftmost square to the last non-blank or previously visited square), m is the position of the read/write head, and q is the current state of the machine.

Definition 5.17 (Initial configuration). The *initial configuration* of M for input $I \in \Sigma^*$ is

$$\langle \rhd \frown I, 1, q_0 \rangle.$$

Definition 5.18 (Yields in one step). We say that a configuration $\langle C, m, q \rangle$ *yields the configuration* $\langle C', m', q' \rangle$ *in one step* (according to M) iff

1. the m-th symbol of C is σ,
2. the instruction set of M specifies $\delta(q, \sigma) = \langle q', \sigma', D \rangle$,
3. the m-th symbol of C' is σ', and
4. a) $D = L$ and $m' = m - 1$ if $m > 0$, otherwise $m' = 0$, or
   b) $D = R$ and $m' = m + 1$, or
   c) $D = N$ and $m' = m$,
5. if $m' = \mathrm{len}(C)$, then $\mathrm{len}(C') = \mathrm{len}(C) + 1$ and the m'-th symbol of C' is 0; otherwise $\mathrm{len}(C') = \mathrm{len}(C)$,
6. for all i such that $i < \mathrm{len}(C)$ and $i \neq m$, $C'(i) = C(i)$.

Definition 5.19 (Run, halting, output). A *run of M on input I* is a sequence C_i of configurations of M, where C_0 is the initial configuration of M for input I, and each C_i yields C_{i+1} in one step.

We say that M *halts on input I after k steps* if $C_k = \langle C, m, q \rangle$, the mth symbol of C is σ, and $\delta(q, \sigma)$ is undefined. In that case, the *output* of M for input I is O, where O is a string of symbols not ending in 0 such that $C = \triangleright \frown O \frown 0^j$ for some $j \in \mathbb{N}$.

5.2.3 Unary representation and computation of functions

We represent natural numbers on the tape using unary notation: if $n \in \mathbb{N}$, let 1^n be the empty sequence if $n = 0$, and otherwise the sequence consisting of exactly n 1's.

Definition 5.20 (Turing computation of a total function)**.** A Turing machine M *computes* the function $f\colon \mathbb{N}^k \to \mathbb{N}$ iff M halts on input

$$1^{n_1}01^{n_2}0\cdots01^{n_k}$$

with output $1^{f(n_1,\ldots,n_k)}$.

Definition 5.21 (Turing computation of a partial function)**.** A Turing machine M computes the partial function $f\colon \mathbb{N}^k \nrightarrow \mathbb{N}$ iff

1. M halts on input $1^{n_1} \frown 0 \frown \cdots \frown 0 \frown 1^{n_k}$ with output 1^m if $f(n_1, \ldots, n_k) = m$.
2. M does not halt at all, or halts with an output that is not a single block of 1's, if $f(n_1, \ldots, n_k)$ is undefined.

5.2.4 Disciplined machines

Definition 5.22 (Disciplined Turing machine)**.** A Turing machine M is *disciplined* iff

1. it has a designated single halting state h,
2. it halts, if it halts at all, while scanning square 1,
3. it never erases the $\triangleright$ symbol on square 0, and
4. it never attempts to move left from square 0.

Proposition 5.23. *For every Turing machine M, there is a disciplined Turing machine M' which halts with output O if M halts with output O, and does not halt if M does not halt. In particular, any function $f\colon \mathbb{N}^n \to \mathbb{N}$ computable by a Turing machine is also computable by a disciplined Turing machine.*

Proof sketch. If M halts in a state other than a designated halting state, add a new state h and transition to it. If M halts with the head not on square 1, add instructions to move the head left until the tape-end marker is found, then move one square right, then halt. The other conditions (not erasing $\triangleright$, not moving left from square 0) can be enforced similarly by adding a bounded number of extra states. □

5.2.5 Combining Turing machines

Given Turing machines $M = \langle Q, \Sigma, q_0, \delta \rangle$ and $M' = \langle Q', \Sigma', q_0', \delta' \rangle$, the *sequential composition $M \frown M'$* is constructed as follows: renumber the states of M' so that $Q \cap Q' = \emptyset$; the states of $M \frown M'$ are $Q \cup Q'$; the alphabet is $\Sigma \cup \Sigma'$; the start state is q_0; and the transition function is

$$\delta''(q, \sigma) = \begin{cases} \delta(q, \sigma) & \text{if } q \in Q \text{ and } \delta(q, \sigma) \text{ is defined} \\ \langle q_0', \sigma, N \rangle & \text{if } q \in Q \text{ and } \delta(q, \sigma) \text{ is undefined} \\ \delta'(q, \sigma) & \text{if } q \in Q'. \end{cases}$$

The idea is that when M would halt (i.e., δ is undefined), the combined machine instead enters the start state of M' and continues.

Proposition 5.24. *If M and M' are disciplined and compute the functions $f\colon \mathbb{N}^k \to \mathbb{N}$ and $f'\colon \mathbb{N} \to \mathbb{N}$, respectively, then $M \frown M'$ is disciplined and computes $f' \circ f$.*

Proof sketch. Since M is disciplined, when it halts with output $f(n_1, \dots, n_k) = m$, the head is scanning square 1. Entering the start state of M' at that point, M' then computes $f'(m)$ and halts on square 1. The other conditions of Definition 5.22 are preserved by the construction. □

5.2.6 The Church–Turing thesis

Definition 5.25 (Church–Turing thesis)**.** The *Church–Turing Thesis* states that anything computable via an effective procedure is Turing computable.

The Church–Turing thesis is supported by the fact that every proposed precise model of effective computability—Turing machines, the λ-calculus, partial recursive functions, register machines, Post production systems, Markov algorithms, and others—turns out to compute exactly the same class of functions. The thesis is invoked in two ways: first, as justification that an informal procedure described in "pseudo-code" could in principle be implemented by a Turing machine; second, and more importantly, to conclude that functions which provably cannot be computed by any Turing machine cannot be computed by *any* effective procedure whatsoever.

Remark 24 (Equivalence of computation models). Turing machines, partial recursive functions, the λ-calculus, unlimited register machines, and all other standard models of computation define exactly the same class of computable (partial) functions. This convergence of independent formalizations constitutes the primary evidence for the Church–Turing thesis.

5.3 Decidability

We now lift the notion of computability from functions to sets and relations. A set is *decidable* (computable) if its characteristic function is computable; it is *semi-decidable* (computably enumerable) if there is a computable procedure that eventually confirms membership for elements of the set, but may fail to terminate for non-members.

5.3.1 Computable sets

Definition 5.26 (Computable / decidable set)**.** Let S be a set of natural numbers. Then S is *computable* (equivalently, *decidable*) iff its characteristic function χ_S is computable, i.e., the function

$$\chi_S(x) = \begin{cases} 1 & \text{if } x \in S \\ 0 & \text{otherwise} \end{cases}$$

is computable. Similarly, a relation $R(x_0, \ldots, x_{k-1})$ is computable iff its characteristic function is computable.

Note the distinction: the computation of a partial function returns the output of the function for input values at which the function is defined; the computation for a decidable set always halts and returns either 1 or 0, indicating membership.

5.3.2 Computably enumerable sets

Definition 5.27 (Computably enumerable set)**.** A set S is *computably enumerable* (abbreviated *c.e.*; also called *recursively enumerable* or *r.e.*) if it is empty or the range of a computable function.

If S is the range of the computable function f, then $S = \{f(0), f(1), f(2), \dots\}$, and f can be seen as "enumerating" the elements of S. The enumeration need not be in increasing order, and repetitions are allowed.

Any computable set is computably enumerable. For if S is computable and non-empty, let a be any element of S and define

$$f(x) = \begin{cases} x & \text{if } \chi_S(x) = 1 \\ a & \text{otherwise.} \end{cases}$$

Then f is computable and S is the range of f.

5.3.3 Equivalent characterizations of c.e. sets

Theorem 5.28. *Let S be a set of natural numbers. Then the following are equivalent:*

1. *S is computably enumerable.*
2. *S is the range of a* partial *computable function.*
3. *S is empty or the range of a primitive recursive function.*
4. *S is the* domain *of a partial computable function.*

Proof. Since every primitive recursive function is computable and every computable function is partial computable, $3 \Rightarrow 1$ and $1 \Rightarrow 2$. (If S is empty, it is the range of the partial computable function that is nowhere defined.) It suffices to show that $2 \Rightarrow 3$ and that $1 \Leftrightarrow 4$.

2 ⇒ 3: Suppose S is the range of the partial computable function φ_e. If S is empty, we are done. Otherwise, let a be any element of S. By Kleene's normal form theorem (DEF-CMP005, §5.4 below),

$$\varphi_e(x) = U(\mu s\, T(e, x, s)).$$

In particular, $\varphi_e(x) \downarrow$ and equals y if and only if there is an s such that $T(e, x, s)$ and $U(s) = y$. Define $f(z)$ by

$$f(z) = \begin{cases} U((z)_1) & \text{if } T(e, (z)_0, (z)_1) \\ a & \text{otherwise.} \end{cases}$$

Then f is primitive recursive, because T and U are. We show S is the range of f. In the forward direction, if $y \in S$, then y is in the range of φ_e, so for some x and s, $T(e, x, s)$ holds and $U(s) = y$; but then $y = f(\langle x, s\rangle)$. Conversely, if y

is in the range of f, then either $y = a \in S$, or for some z, $T(e, (z)_0, (z)_1)$ and $U((z)_1) = y$; in the latter case $\varphi_e((z)_0) \downarrow = y$, so $y \in S$.

1 ⇒ 4: Suppose S is the range of a computable function f, i.e., $S = \{y :$ for some x, $f(x) = y\}$. Let

$$g(y) = \mu x\ (f(x) = y).$$

Then g is a partial computable function, and $g(y)$ is defined if and only if for some x, $f(x) = y$. So the domain of g is the range of f, which is S.

4 ⇒ 1: Suppose S is the domain of the partial computable function φ_e, i.e., $S = \{x : \varphi_e(x) \downarrow\}$. If S is empty, we are done; otherwise, let a be any element of S. Define f by

$$f(z) = \begin{cases} (z)_0 & \text{if } T(e, (z)_0, (z)_1) \\ a & \text{otherwise.} \end{cases}$$

Then a number x is in the range of f if and only if $\varphi_e(x) \downarrow$, i.e., if and only if $x \in S$. □

Clause 4 provides a convenient way of enumerating the c.e. sets: for each e, let W_e denote the domain of φ_e, i.e.,

$$W_e = \{x : \varphi_e(x) \downarrow\}.$$

Then if A is any computably enumerable set, $A = W_e$ for some e.

Theorem 5.29 (∃-characterization of c.e. sets). *A set S is computably enumerable if and only if there is a computable relation $R(x, y)$ such that*

$$S = \{x : \exists y\, R(x, y)\}.$$

Proof. In the forward direction, suppose S is computably enumerable. Then for some e, $S = W_e$. For this value of e we can write

$$S = \{x : \exists y\, T(e, x, y)\}.$$

In the reverse direction, suppose $S = \{x : \exists y\, R(x, y)\}$. Define f by

$$f(x) \simeq \mu y\ R(x, y).$$

Then f is partial computable, and S is the domain of f. □

5.3.4 Closure properties of c.e. sets

Theorem 5.30. *Suppose A and B are computably enumerable. Then so are $A \cap B$ and $A \cup B$.*

Proof sketch. Suppose A is the domain of φ_d and B is the domain of φ_e. Then $A \cap B$ is the domain of the partial function $\varphi_d(x) + \varphi_e(x)$ (both must halt for the sum to be defined). For $A \cup B$, define $p(x) = \mu y\ (T(d, x, y) \vee T(e, x, y))$; then $A \cup B$ is the domain of p, since p halts whenever either φ_d or φ_e halts. □

5.3.5 C.e. sets are not closed under complement

Theorem 5.31. *Let A be any set of natural numbers. Then A is computable if and only if both A and $\overline{A}$ are computably enumerable.*

Proof. The forward direction is straightforward: if A is computable, then $\overline{A}$ is also computable (since $\chi_{\overline{A}} = 1 \dot{-} \chi_A$), and so both are c.e.

In the reverse direction, suppose A and $\overline{A}$ are both computably enumerable. Let A be the domain of φ_d, and let $\overline{A}$ be the domain of φ_e. Define h by

$$h(x) = \mu s\ (T(d, x, s) \vee T(e, x, s)).$$

On input x, h searches for either a halting computation of φ_d or a halting computation of φ_e. Since every x is in either A or $\overline{A}$, one of these searches must succeed, so h is total computable. Now for every x: $x \in A$ if and only if $T(d, x, h(x))$, i.e., if φ_d is the one that halts. Since $T(d, x, h(x))$ is a computable relation, A is computable. □

5.3.6 Non-computable sets

Theorem 5.32. *Let $K_0 = \{\langle e, x\rangle : \varphi_e(x) \downarrow\}$. Then K_0 is computably enumerable but not computable.*

Proof. To see that K_0 is computably enumerable, note that it is the domain of the function f defined by

$$f(z) = \mu y\ (\mathrm{len}(z) = 2 \wedge T((z)_0, (z)_1, y)).$$

For, if $\varphi_e(x)$ is defined, $f(\langle e, x\rangle)$ finds a halting computation sequence; if $\varphi_e(x)$ is undefined, so is $f(\langle e, x\rangle)$; and if z doesn't code a pair, then $f(z)$ is also undefined.

The fact that K_0 is not computable is the undecidability of the halting problem: if K_0 were decidable, one could decide for any program e and input x whether $\varphi_e(x)$ halts, contradicting the halting problem (see §5.4). □

The set K_0 is the *halting set*: $\langle e, x\rangle \in K_0$ iff φ_e is defined on input x.

Theorem 5.33. *The* self-halting set $K = \{e : \varphi_e(e) \downarrow\}$ *is computably enumerable but not decidable.*

Proof. Suppose K is decidable, i.e., its characteristic function χ_K is computable. Define

$$d(e) = \begin{cases} 1 & \text{if } \chi_K(e) = 0 \\ \uparrow & \text{otherwise.} \end{cases}$$

Let k be the index of d, i.e., $d \simeq \varphi_k$. Then $d(k) \simeq \varphi_k(k)$. But by definition, $d(k) \downarrow$ iff $\chi_K(k) = 0$ iff $k \notin K$ iff $\varphi_k(k) \uparrow$—a contradiction.

K is c.e. because it is the domain of $f(x) = \mu y\ T(x, x, y)$. □

Corollary 5.34. $\overline{K_0}$ *is not computably enumerable.*

Proof. We know that K_0 is computably enumerable but not computable. If $\overline{K_0}$ were computably enumerable, then K_0 would be computable by Theorem 5.31, contradicting Theorem 5.32. □

5.4 Diagonalization and Halting

Diagonalization is one of the most powerful techniques in the theory of computation. It was first used by Cantor to show that the set of real numbers is uncountable, and it was adapted by Gödel and Turing to establish fundamental limits on what can be computed. Several results in §5.3 (e.g., the noncomputability of K_0) were stated using the halting problem; we now give the detailed proofs. We use diagonalization to prove that the halting problem is unsolvable, and that there is no universal computable function for the total computable functions.

5.4.1 No universal computable function

Although there is a partial computable function that is universal for the partial computable functions (see Theorem 5.45 below), there is no total computable function that is universal for the total computable functions.

Theorem 5.35. *There is no universal computable function. In other words, any function* $\mathrm{Un}'(k, x)$ *which is such that if* $f(x)$ *is a total computable function, then there is a natural number* k *such that* $f(x) = \mathrm{Un}'(k, x)$ *for every* x*, is not computable.*

Proof. The proof is a simple diagonalization: if $\mathrm{Un}'(k, x)$ were total and computable, then

$$d(x) = \mathrm{Un}'(x, x) + 1$$

would also be total and computable. However, by definition, $d(k)$ is not equal to $\mathrm{Un}'(k,k)$. Hence, for every k, the values of $d(x)$ and $\mathrm{Un}'(k,x)$ differ for at least one x, namely $x = k$. $\square$

The normal form theorem (see Theorem 5.41) shows that we can get around this diagonalization argument, but only at the expense of allowing the universal function to be partial. That is, Un is universal for the total computable functions, it just isn't total. The diagonalization argument does not work in the partial case.

Remark 25 (Diagonalization for primitive recursive functions). The same technique shows that the primitive recursive functions do not exhaust the computable functions. One can effectively enumerate all unary primitive recursive functions $f_0, f_1, f_2, \ldots$ (by assigning codes to their definitions; see PRIM-CMP011, §5.5). The function $h(x) = f_x(x) + 1$ is then computable but not primitive recursive, since it differs from each f_i at argument i.

5.4.2 The halting problem

Assume we have fixed an enumeration of Turing machine descriptions $M_1, M_2, M_3, \ldots$ (see Definition 5.52). Each Turing machine thus receives an *index*: its place in the enumeration. We know that there must be non-Turing-computable functions—the set of Turing machine descriptions is enumerable, but the set of all functions from $\mathbb{N}$ to $\mathbb{N}$ is not. But we can find specific examples of non-computable functions.

Definition 5.36 (Halting function)**.** The *halting function h* is defined as

$$h(e,n) = \begin{cases} 0 & \text{if machine } M_e \text{ does not halt for input } n \\ 1 & \text{if machine } M_e \text{ halts for input } n \end{cases}$$

Definition 5.37 (Halting problem)**.** The *Halting Problem* is the problem of determining (for any e, n) whether the Turing machine M_e halts for an input of n strokes.

We show that h is not Turing-computable by showing that a related function s is not Turing-computable. This proof relies on the fact that anything computable by a Turing machine can be computed by a disciplined Turing machine (Definition 5.22), and the fact that two Turing machines can be combined into a single machine (Proposition 5.24).

Definition 5.38. The function s is defined as

$$s(e) = \begin{cases} 0 & \text{if machine } M_e \text{ does not halt for input } e \\ 1 & \text{if machine } M_e \text{ halts for input } e \end{cases}$$

Lemma 5.39. *The function s is not Turing computable.*

Proof. We suppose, for contradiction, that the function s is Turing computable. Then there would be a Turing machine S that computes s. We may assume, without loss of generality, that when S halts, it does so while scanning the first square (i.e., that it is disciplined). This machine can be "hooked up" to another machine J, which halts if it is started on input 0 (i.e., if it reads 0 in the initial state while scanning the square to the right of the end-of-tape symbol), and otherwise wanders off to the right, never halting. $S \frown J$, the machine created by hooking S to J, is a Turing machine, so it is M_e for some e (i.e., it appears somewhere in the enumeration). Start M_e on an input of e 1s. There are two possibilities: either M_e halts or it does not halt.

1. Suppose M_e halts for an input of e 1s. Then $s(e) = 1$. So S, when started on e, halts with a single 1 as output on the tape. Then J starts with a 1 on the tape. In that case J does not halt. But M_e is the machine $S \frown J$, so it should do exactly what S followed by J would do (i.e., in this case, wander off to the right and never halt). So M_e cannot halt for an input of e 1's.

2. Now suppose M_e does not halt for an input of e 1s. Then $s(e) = 0$, and S, when started on input e, halts with a blank tape. J, when started on a blank tape, immediately halts. Again, M_e does what S followed by J would do, so M_e must halt for an input of e 1's.

In each case we arrive at a contradiction with our assumption. This shows there cannot be a Turing machine S: s is not Turing computable. □

Theorem 5.40 (Unsolvability of the Halting Problem)**.** *The halting problem is unsolvable, i.e., the function h is not Turing computable.*

Proof. Suppose h were Turing computable, say, by a Turing machine H. We could use H to build a Turing machine that computes s: First, make a copy of the input (separated by a 0 symbol). Then move back to the beginning, and run H. We can clearly make a machine that does the former, and if H existed, we would be able to "hook it up" to such a copier machine to get a new machine which would determine if M_e halts on input e, i.e., computes s. But we've already shown that no such machine can exist. Hence, h is also not Turing computable. $\square$

Remark 26 (Halting problem for partial recursive functions). The same result holds in the setting of partial recursive functions. Let $\mathrm{Un}(e, x)$ denote the universal partial computable function. Define

$$h(e,n) = \begin{cases} 1 & \text{if } \mathrm{Un}(e,n) \text{ is defined} \\ 0 & \text{otherwise.} \end{cases}$$

Then h is not computable. One proof goes via the no-universal-function result (Theorem 5.35): if h were computable, one could define a total computable $\mathrm{Un}'(e,n)$ agreeing with Un wherever the latter is defined (by returning 0 when $h(e,n) = 0$), contradicting Theorem 5.35. An alternative, more direct proof proceeds by diagonalization: define $g(n)$ to equal 0 when $h(n,n) = 0$ and be undefined otherwise. Then g is partial computable, so $g \simeq \varphi_e$ for some e, and asking whether $g(e)$ is defined leads to a contradiction.

5.5 Coding and Universality

We now develop the machinery of Gödel numbering, which allows us to treat syntactic objects—terms, formulas, derivations, and programs—as natural numbers. This opens the door to the normal form theorem, the universal Turing machine, the *s-m-n* theorem, arithmetization of proof predicates, and the representability theorem.

5.5.1 Sequence coding

The set of primitive recursive functions is remarkably robust, and we can extend its power further with a coding of finite sequences of natural numbers as single natural numbers. We identify the sequence $\langle a_0, a_1, \ldots, a_k \rangle$ with the number

$$p_0^{a_0+1} \cdot p_1^{a_1+1} \cdot p_2^{a_2+1} \cdot \cdots \cdot p_k^{a_k+1},$$

where p_i is the ith prime. Adding one to the exponents ensures that, e.g., $\langle 2,7,3 \rangle$ and $\langle 2,7,3,0,0 \rangle$ have distinct codes. The Fundamental Theorem of Arithmetic guarantees that this mapping is injective.

The operations of determining the length $\mathrm{len}(s)$ of a sequence s, extracting its ith element $(s)_i$, appending an element $\mathrm{append}(s,a)$, and concatenating two sequences $s \frown t$, are all primitive recursive. We can also bound the code of a sequence of length k with elements at most x by $\mathrm{sequenceBound}(x,k) = p_{k-1}^{k\cdot(x+1)}$.

5.5.2 Kleene's normal form theorem

Theorem 5.41 (Kleene's Normal Form Theorem)**.** *There is a primitive recursive relation $T(e,x,s)$ and a primitive recursive function $U(s)$, with the following property: if f is any partial recursive function, then for some e,*

$$f(x) \simeq U(\mu s\ T(e,x,s))$$

for every x.

Every partial recursive function has an *index e*—intuitively, a number coding its program or definition. If $f(x)\downarrow$, the computation can be recorded and coded by some number s, and the fact that s codes the computation of f on input x can be checked primitive recursively. Consequently, $T(e,x,s)$ ("the function with index e has a computation for input x coded by s") is primitive recursive, and the output can be extracted from s by the primitive recursive function U.

Definition 5.42 (Index / program)**.** The normal form theorem shows that only a single unbounded search is required for the definition of any partial recursive function. We use the numbers e as "names" of partial recursive functions, and write φ_e for the function f defined by the equation $\varphi_e(x) \simeq U(\mu s\ T(e,x,s))$. Note that any partial recursive function can have more than one index—in fact, every partial recursive function has infinitely many indices.

5.5.3 Enumerating Turing machines

Every Turing machine can be described by a finite sequence of positive integers encoding its states, alphabet, start state, and instructions. By considering only *standard* machines (where states and symbols are positive integers), we can canonically encode any Turing machine as an element of $(\mathbb{Z}^+)^*$. Since $(\mathbb{Z}^+)^*$ is enumerable, so is the set of standard Turing machine descriptions.

Definition 5.43 (Index of a Turing machine)**.** If M is the eth Turing machine (in our fixed enumeration), we say that e is an *index* of M. We write M_e for the eth Turing machine.

A machine may have more than one index; for example, two descriptions of M that list the instructions in different orders will have different indices. Given the enumeration, we can effectively compute the description of M from its index and vice versa.

Theorem 5.44. *There are functions from $\mathbb{N}$ to $\mathbb{N}$ which are not Turing computable.*

Proof. The set of descriptions of standard Turing machines is a subset of $(\mathbb{Z}^+)^*$, so it is enumerable. The set of all Turing computable functions is therefore also enumerable. But the set of all functions from $\mathbb{N}$ to $\mathbb{N}$ is not enumerable. So there must be functions that are not Turing computable. □

5.5.4 The universal Turing machine

Theorem 5.45 (Universal Turing machine)**.** *There is a* universal Turing machine U *which, when started on input* $\langle e, n \rangle$*:*

1. *halts iff M_e halts on input n, and*
2. *if M_e halts with output m, so does U.*

U thus computes the function $f\colon \mathbb{N} \times \mathbb{N} \nrightarrow \mathbb{N}$ given by $f(e, n) = m$ if M_e started on input n halts with output m, and undefined otherwise.

Proof. We describe how U works and invoke the Church–Turing thesis. When U starts, its tape contains a block of e 1's followed by a block of n 1's. It first "decodes" the index e, producing the description of M_e (a list of instruction 5-tuples). Then U sets up the initial configuration: it records the start state of M_e and the initial head position, and converts the input into coded symbols on a simulated "tape."

U now simulates M_e step by step:

1. Find the current head position k.
2. Read the coded symbol at position k on the simulated tape.
3. Find the instruction matching the current state and symbol.
4. Write the new symbol at position k.
5. Update the stored state to the new state.
6. Adjust the stored head position (increment or decrement according to the direction).
7. Repeat.

If M_e never halts, then U never halts either. If M_e halts (i.e., no instruction matches the current state/symbol pair), then U decodes the simulated tape contents back into a unary output and halts. □

5.5.5 The s-m-n theorem

Theorem 5.46 (s-m-n theorem)**.** *For each pair of natural numbers n and m, there is a primitive recursive function s^m_n such that for every sequence e, a_0, …, a_{m-1}, y_0, …, y_{n-1}, we have*

$$\varphi^n_{s^m_n(e,a_0,\ldots,a_{m-1})}(y_0,\ldots,y_{n-1}) \simeq \varphi^{m+n}_e(a_0,\ldots,a_{m-1},y_0,\ldots,y_{n-1}).$$

It is helpful to think of s^m_n as acting on *programs*. The function s^m_n takes a program e for an $(m+n)$-ary function, as well as fixed inputs $a_0,\ldots,a_{m-1}$, and returns a program $s^m_n(e,a_0,\ldots,a_{m-1})$ for the n-ary function of the remaining arguments. In Turing machine terms, $s^m_n(e,a_0,\ldots,a_{m-1})$ is the machine that, on input $y_0,\ldots,y_{n-1}$, prepends $a_0,\ldots,a_{m-1}$ to the input string and runs e. Each s^m_n is a primitive recursive function that finds a code for the appropriate machine.

5.5.6 Representing Turing machines in first-order logic

To connect computation to logic, we show how to represent the behavior of a Turing machine M on input w by a sentence of first-order logic.

Definition 5.47 (Language $\mathcal{L}_M$)**.** Given a Turing machine $M = \langle Q, \Sigma, q_0, \delta\rangle$, the language $\mathcal{L}_M$ consists of:

1. A two-place predicate symbol $Q_q(x,y)$ for every state $q \in Q$. Intuitively, $Q_q(\overline{m},\overline{n})$ expresses "after n steps, M is in state q scanning the mth square."
2. A two-place predicate symbol $S_\sigma(x,y)$ for every symbol $\sigma \in \Sigma$. Intuitively, $S_\sigma(\overline{m},\overline{n})$ expresses "after n steps, the mth square contains symbol σ."
3. A constant symbol 0, a one-place function symbol $'$, and a two-place predicate symbol $<$.

The sentence $\tau(M,w)$ consists of axioms for 0, $'$, and $<$ (ensuring $\forall x\, x < x'$ and transitivity), axioms describing the input configuration, and axioms describing the transition from one configuration to the next. For each instruction $\delta(q_i,\sigma) = \langle q_j,\sigma',D\rangle$ there is a universally quantified sentence stating that if M is in state q_i scanning a square containing σ, then after one more step, the state is q_j, the symbol on that square is σ', the head has moved according to D, and all other squares are unchanged.

The sentence $\alpha(M,w)$ asserts that M eventually reaches a halting configuration: $\exists x\, \exists y\, (\bigvee_{\langle q,\sigma\rangle\in X}(Q_q(x,y) \land S_\sigma(x,y)))$, where X is the set of state/symbol pairs on which δ is undefined.

5.5.7 Verification lemmas

Let $\chi(M, w, n)$ be the sentence describing the configuration of M run on w after n steps: it specifies the state, head position, and contents of every tape square.

Lemma 5.48. *If M run on input w is in a halting configuration after n steps, then $\chi(M, w, n) \vDash \alpha(M, w)$.*

Proof sketch. If M halts after n steps in state q scanning square m containing σ with $\delta(q, \sigma)$ undefined, then $\chi(M, w, n)$ includes conjuncts $Q_q(\overline{m}, \overline{n})$ and $S_\sigma(\overline{m}, \overline{n})$. Since $\langle q, \sigma \rangle \in X$, these imply $\alpha(M, w)$ by existential generalization. □

Lemma 5.49. *For each n, if M has not halted after n steps, $\tau(M, w) \vDash \chi(M, w, n)$.*

Proof sketch. By induction on n. The base case ($n = 0$) holds because the conjuncts of $\chi(M, w, 0)$ are conjuncts of $\tau(M, w)$. For the inductive step, suppose $\tau(M, w) \vDash \chi(M, w, n)$ and M has not halted. The transition axiom corresponding to the instruction executed at step n, together with $\chi(M, w, n)$, entails all conjuncts of $\chi(M, w, n+1)$. Unchanged squares follow from the frame axiom $\varphi(x, y)$; if the head visits a new square, the axiom $\forall x\, x < x'$ and transitivity establish the required properties. □

Lemma 5.50. *If M halts on input w, then $\tau(M, w) \to \alpha(M, w)$ is valid.*

Proof sketch. If M halts after k steps, then by Lemma 5.49, $\tau(M, w) \vDash \chi(M, w, k)$, and by Lemma 5.48, $\chi(M, w, k) \vDash \alpha(M, w)$. □

Lemma 5.51. *If $\vDash \tau(M, w) \to \alpha(M, w)$, then M halts on input w.*

Proof sketch. Construct a structure $\mathfrak{M}$ with domain $\mathbb{N}$ that interprets o as 0, $'$ as successor, $<$ as less-than, and Q_q, S_σ according to the actual run of M on w. Then $\mathfrak{M} \vDash \tau(M, w)$ by construction. If $\vDash \tau(M, w) \to \alpha(M, w)$, then $\mathfrak{M} \vDash \alpha(M, w)$, which means there exist $m, n \in \mathbb{N}$ such that M is in a halting configuration after n steps. □

5.5.8 Arithmetization of syntax

We now show that syntactic objects of first-order logic can be coded as natural numbers, and that the relevant properties are primitive recursive.

Definition 5.52 (Symbol code and Gödel number)**.** If s is a symbol of the language $\mathcal{L}$, its *symbol code* c_s is defined as follows: logical symbols receive codes $\langle 0, i\rangle$ for various i; the ith variable v_i receives $\mathrm{c}_{v_i} = \langle 1, i\rangle$; the ith constant symbol c_i receives $\langle 2, i\rangle$; the ith n-ary function symbol f^n_i receives $\langle 3, n, i\rangle$; and the ith n-ary predicate symbol P^n_i receives $\langle 4, n, i\rangle$. If $s_0, \dots, s_{n-1}$ is a sequence of symbols, its *Gödel number* is $\langle \mathrm{c}_{s_0}, \dots, \mathrm{c}_{s_{n-1}}\rangle$.

Remark 27. The relations $\mathrm{Fn}(x, n)$ ("x codes an n-ary function symbol") and $\mathrm{Pred}(x, n)$ ("x codes an n-ary predicate symbol") are primitive recursive.

The following properties are all primitive recursive (each is established by bounded search over formation sequences or codes):

Proposition 5.53. *The relation* $\mathrm{Term}(x)$*, which holds iff x is the Gödel number of a term, is primitive recursive. So is* $\mathrm{num}(n) = {}^{\#}\overline{n}^{\#}$.

Proof sketch. A number x is the Gödel number of a term iff there is a formation sequence $s_0, \dots, s_{k-1}$ of terms ending in the expression coded by x. Each s_i is either a variable, a constant, or is built from earlier terms by a function symbol. The code of the formation sequence is bounded by $p_{k-1}^{k(x+1)}$ where $k = \mathrm{len}(x)$, so the check involves only bounded quantification. The function $\mathrm{num}(n)$ is defined by primitive recursion: $\mathrm{num}(0) = {}^{\#}\mathrm{o}^{\#}$ and $\mathrm{num}(n+1) = {}^{\#}{\prime}({}^{\#} \frown \mathrm{num}(n) \frown {}^{\#}){}^{\#}$. □

Proposition 5.54. *The relations* $\mathrm{Frm}(x)$ *("x is the Gödel number of a formula") and* $\mathrm{Sent}(x)$ *("x is the Gödel number of a sentence") are primitive recursive.*

Proposition 5.55. *There is a primitive recursive function* $\mathrm{Subst}(x, y, z)$ *such that* $\mathrm{Subst}({}^{\#}\varphi^{\#}, {}^{\#}t^{\#}, {}^{\#}u^{\#}) = {}^{\#}\varphi[t/u]^{\#}$.

5.5.9 The proof predicate

We now arithmetize derivations. Since derivations are structured syntactic objects (trees of sequents or sequences of formulas), they can be coded as numbers. The details depend on the proof system chosen—sequent calculus (**LK**), natural deduction, or axiomatic derivations—but the result is the same in each case.

Definition 5.56 (Gödel number of a derivation)**.** A derivation π receives a Gödel number ${}^{\#}\pi^{\#}$ by recursively coding its structure. For sequent calculus: an initial sequent $\Gamma \Rightarrow \Delta$ is coded as $\langle 0, {}^{\#}\Gamma \Rightarrow \Delta^{\#}\rangle$; a one-premise inference is coded as $\langle 1, {}^{\#}\pi_1{}^{\#}, {}^{\#}\Gamma \Rightarrow \Delta^{\#}, k\rangle$ where k identifies the rule; and similarly for two-premise inferences. Analogous codings exist for natural deduction and axiomatic derivations.

Proposition 5.57. *The property* $\mathrm{Correct}(p)$*, which holds iff the last inference in the derivation with Gödel number p is a correct application of a rule, is primitive recursive.*

Proof sketch. For each rule R, the relation $\mathrm{FollowsBy}_R(p)$ checks that the end-sequent of p follows from the premises by a correct application of R. This involves verifying the structure of the Gödel numbers using bounded quantification and the primitive recursive functions for sequence manipulation, Frm, Subst, etc. The property $\mathrm{Correct}(p)$ is the disjunction of all $\mathrm{FollowsBy}_R(p)$ together with the case that p codes an initial sequent. □

Proposition 5.58. *The relation* $\mathrm{Deriv}(p)$*, which holds if p is the Gödel number of a correct derivation, is primitive recursive.*

Proposition 5.59 (Proof predicate)**.** *Suppose Γ is a primitive recursive set of sentences. Then the relation* $\mathrm{Prf}_\Gamma(x, y)$ *expressing "x is the code of a derivation of φ from Γ and y is the Gödel number of φ" is primitive recursive.*

Remark 28 (Variant proof systems). The above results hold for sequent calculus, natural deduction, and axiomatic proof systems. The internal details of the coding differ (trees vs. sequences, treatment of discharge labels, etc.), but in every case Deriv and Prf_Γ are primitive recursive. The key ingredients are the same: primitive recursive checking of each inference step, and bounded search over sub-derivations.

5.5.10 Representability in Q

Definition 5.60 (Representable function)**.** A function $f(x_0, \ldots, x_k)$ is *representable in* **Q** if there is a formula $\varphi_f(x_0, \ldots, x_k, y)$ such that whenever $f(n_0, \ldots, n_k) = m$, then:

1. $\mathbf{Q} \vdash \varphi_f(\overline{n_0}, \dots, \overline{n_k}, \overline{m})$, and
2. $\mathbf{Q} \vdash \forall y\, (\varphi_f(\overline{n_0}, \dots, \overline{n_k}, y) \rightarrow y = \overline{m})$.

Definition 5.61 (Representable relation)**.** A relation $R(x_0, \dots, x_k)$ is *representable in* $\mathbf{Q}$ if there is a formula $\varphi_R(x_0, \dots, x_k)$ such that whenever $R(n_0, \dots, n_k)$ is true, $\mathbf{Q} \vdash \varphi_R(\overline{n_0}, \dots, \overline{n_k})$, and whenever $R(n_0, \dots, n_k)$ is false, $\mathbf{Q} \vdash \neg\varphi_R(\overline{n_0}, \dots, \overline{n_k})$.

The representability theorem establishes a deep connection between computability and provability.

Theorem 5.62 (Representability theorem)**.** *A function is representable in* $\mathbf{Q}$ *if and only if it is computable. A relation is representable in* $\mathbf{Q}$ *if and only if it is computable.*

Proof sketch. **Representable $\Rightarrow$ computable:** If f is represented by φ_f, we compute $f(n_0, \dots, n_k)$ by searching through all derivations from $\mathbf{Q}$ until we find one proving $\varphi_f(\overline{n_0}, \dots, \overline{n_k}, \overline{m})$ for some m. Since the proof predicate $\mathrm{Prf}_{\mathbf{Q}}$ is primitive recursive, the search can be formalized as a regular minimization.

Computable $\Rightarrow$ representable: We show that the basic functions (zero, succ, P^n_i, add, mult, the equality relation $=$) are representable, and that representable functions are closed under composition and regular minimization. Primitive recursion is handled via the beta function, as follows.

The basic functions are represented by their natural formulas: zero by $y = 0$, succ by $y = x'$, P^n_i by $y = x_i$, add by $y = (x_0 + x_1)$, mult by $y = (x_0 \times x_1)$, and the equality relation $=$ by $(x_0 = x_1 \land y = \overline{1}) \lor (x_0 \neq x_1 \land y = \overline{0})$. That these work requires showing, for instance, that $\mathbf{Q} \vdash (\overline{n} + \overline{m}) = \overline{n+m}$ (by induction on m using axioms Q_4 and Q_5) and that $\mathbf{Q}$ proves distinct numerals unequal (by induction using axioms Q_1 and Q_2).

Closure under composition: if φ_f represents f and φ_{g_i} represents g_i, then $\exists y_0 \dots \exists y_{k-1}\, (\varphi_{g_0}(\vec{x}, y_0) \land \dots \land \varphi_{g_{k-1}}(\vec{x}, y_{k-1}) \land \varphi_f(y_0, \dots, y_{k-1}, z))$ represents $h(\vec{x}) = f(g_0(\vec{x}), \dots, g_{k-1}(\vec{x}))$.

Closure under regular minimization: if φ_g represents g, then $\varphi_g(y, z, 0) \land \forall w\, (w < y \rightarrow \neg\varphi_g(w, z, 0))$ represents $f(z) = \mu x\, [g(x, z) = 0]$. The proof uses lemmas showing that $\mathbf{Q}$ proves $\forall x\, \neg x < 0$, that $\mathbf{Q}$ proves $x < \overline{n+1} \rightarrow (x = 0 \lor \dots \lor x = \overline{n})$, and that $\mathbf{Q}$ proves trichotomy for numerals. □

5.5.11 The beta function

Lemma 5.63 (Beta function lemma)**.** *There is a function $\beta(d, i)$ such that for every sequence $a_0, \dots, a_n$ there is a number d such that for every $i \leq n$, $\beta(d, i) = a_i$. Moreover, β can be defined from the basic functions using just composition and regu-*

lar minimization.

The function β provides a way of decoding finite sequences without using primitive recursion. It is defined using the Chinese Remainder Theorem (Sunzi's Theorem): given $a_0, \ldots, a_n$, let $j = \max(n, a_0 + 1, \ldots, a_n + 1)$ and $m = \operatorname{lcm}(1, \ldots, j)$. Then $x_i = 1 + (i+1) \cdot m$ are pairwise relatively prime and each exceeds a_i. By Sunzi's Theorem, there exists d_0 with $d_0 \equiv a_i \pmod{x_i}$ for each i. Setting $d = J(d_0, m)$ (where J is the pairing function) and $\beta(d, i) = \operatorname{rem}(1 + (i+1) \cdot L(d), K(d))$ gives the required decoding.

Using β, primitive recursion can be simulated by regular minimization: if $h(\vec{x}, 0) = f(\vec{x})$ and $h(\vec{x}, y+1) = g(\vec{x}, y, h(\vec{x}, y))$, then $h(\vec{x}, y) = \beta(\hat{h}(\vec{x}, y), y)$ where $\hat{h}(\vec{x}, y) = \mu d\ (\beta(d, 0) = f(\vec{x}) \wedge (\forall i < y)\ \beta(d, i+1) = g(\vec{x}, i, \beta(d, i)))$.

5.5.12 Productive sets

Definition 5.64 (Productive set)**.** A set $A \subseteq \mathbb{N}$ is *productive* if there exists a computable function f such that for every c.e. set $W_e \subseteq A$, we have $f(e) \in A \setminus W_e$. Such a function f is called a *productive function* for A.

Productive sets witness a strong form of non-computability: not only is a productive set not c.e., but given any c.e. approximation $W_e \subseteq A$, we can computably find a specific element that W_e misses.

Proposition 5.65. *The complement of $K = \{e : \varphi_e(e)\downarrow\}$ is productive, with the identity function as a productive function.*

Proof. Suppose $W_e \subseteq \overline{K}$. We must show $e \in \overline{K} \setminus W_e$. If $e \in W_e$, then since $W_e \subseteq \overline{K}$, we have $e \in \overline{K}$, i.e., $\varphi_e(e)\uparrow$. But $e \in W_e$ means $\varphi_e(e)\downarrow$, a contradiction. So $e \notin W_e$. If $e \in K$, then $\varphi_e(e)\downarrow$, so $e \in W_e$ (since W_e is the domain of φ_e), contradicting what we just showed. So $e \in \overline{K}$, and thus $e \in \overline{K} \setminus W_e$. □

5.6 Computability Theory

Having established the basic framework of computability—recursive functions, Turing machines, decidability, coding, and universality—we now develop the structural theory. The key tools are many-one reducibility, which provides a method for comparing the difficulty of decision problems, and the notions of complete c.e. sets, axiomatizable theories, and computable inseparability.

5.6.1 Many-one reducibility

Definition 5.66 (Many-one reduction)**.** Let A and B be sets of natural numbers. A computable function $f\colon \mathbb{N} \to \mathbb{N}$ is a *many-one reduction* of A to B iff, for every natural number x,

$$x \in A \quad \text{if and only if} \quad f(x) \in B.$$

If such a reduction f exists, we say that A is *many-one reducible* to B, written $A \leq_m B$. If $A \leq_m B$ and $B \leq_m A$, then A and B are *many-one equivalent*, written $A \equiv_m B$.

As an example, the function $f(x) = \langle x, x \rangle$ is a many-one reduction of $K = \{x : \varphi_x(x) \downarrow\}$ to $K_0 = \{\langle e, x \rangle : \varphi_e(x) \downarrow\}$, since $x \in K$ iff $\varphi_x(x) \downarrow$ iff $\langle x, x \rangle \in K_0$.

If f happens to be injective, A is said to be *one-one reducible* to B.

Proposition 5.67 (Transitivity)**.** *If $A \leq_m B$ and $B \leq_m C$, then $A \leq_m C$.*

Proof. Composing a reduction of A to B with a reduction of B to C yields a reduction of A to C: if f reduces A to B and g reduces B to C, then $g \circ f$ reduces A to C, since for every x, $x \in A$ iff $f(x) \in B$ iff $g(f(x)) \in C$. □

Proposition 5.68 (Preservation under reduction)**.** *Let A and B be any sets, and suppose $A \leq_m B$.*

1. *If B is computably enumerable, so is A.*
2. *If B is computable, so is A.*

Proof. Let f be a many-one reduction from A to B.

For (1): if B is the domain of a partial computable function g, then A is the domain of $g \circ f$, since $x \in A$ iff $f(x) \in B$ iff $g(f(x)) \downarrow$.

For (2): note that f is also a reduction of $\overline{A}$ to $\overline{B}$. If B is computable, then both B and $\overline{B}$ are c.e., so by (1), both A and $\overline{A}$ are c.e., whence A is computable by Theorem 5.31. □

5.6.2 Complete c.e. sets

Definition 5.69 (Complete c.e. set)**.** A set A is a *complete computably enumerable set* (under many-one reducibility) if

1. A is computably enumerable, and

2. for any other computably enumerable set B, $B \leq_m A$.

In other words, complete c.e. sets are the "hardest" c.e. sets possible: they allow one to answer questions about *any* c.e. set.

A further variant of the halting set is $K_1 = \{e : \varphi_e(0) \downarrow\}$, which records those indices whose program halts on input 0.

Theorem 5.70. *K, K_0, and K_1 are all complete computably enumerable sets.*

Proof. To see that K_0 is complete, let B be any computably enumerable set. Then for some index e, $B = W_e = \{x : \varphi_e(x) \downarrow\}$. Let f be the function $f(x) = \langle e, x \rangle$. Then for every natural number x, $x \in B$ if and only if $f(x) \in K_0$. In other words, f reduces B to K_0.

K can be reduced to K_0 in the same way (via $x \mapsto \langle x, x \rangle$), so K is also complete by transitivity.

To see that K_1 is complete, one shows that K_0 reduces to K_1 (via an *s-m-n* argument), and then completeness follows by transitivity. □

5.6.3 Totality is undecidable

Proposition 5.71. *The set* $\mathrm{Tot} = \{x : \textit{for every } y,\ \varphi_x(y) \downarrow\}$ *is not computable.*

Proof sketch. We reduce K to Tot. Define $h(x, y) \simeq 0$ if $x \in K$, and $h(x, y) \uparrow$ otherwise (the function h does not depend on y: it simply simulates $\varphi_x(x)$ and outputs 0 if it halts). By the *s-m-n* theorem, there is a primitive recursive $k(x)$ such that $\varphi_{k(x)}(y) = h(x, y)$. Then $\varphi_{k(x)}$ is total iff $x \in K$, so k reduces K to Tot. □

5.6.4 Axiomatizable theories

Definition 5.72 (Axiomatizable theory)**.** A theory Γ is *axiomatizable* if it is axiomatized by a decidable set of axioms, i.e., $\Gamma = \{\varphi : \Gamma_0 \vDash \varphi\}$ for some decidable set Γ_0.

Any theory with a finite set of axioms is axiomatizable (since finite sets are decidable). Schematically axiomatized theories like Peano arithmetic **PA** are also axiomatizable, since one can effectively test whether a given sentence is an instance of the induction schema.

Lemma 5.73. *If* **T** *is axiomatizable, then* **T** *is computably enumerable.*

Proof sketch. If A is a decidable set of axioms for **T**, then $\varphi \in \mathbf{T}$ iff there is a finite list of axioms $\psi_1, \dots, \psi_k$ in A and a derivation of $(\psi_1 \wedge \cdots \wedge \psi_k) \to \varphi$ in first-order logic. Since the set of all derivations is enumerable and we can check membership in A decidably, **T** is c.e. □

5.6.5 Computable inseparability

Definition 5.74 (Computable inseparability)**.** Two disjoint sets A and B are *computably inseparable* if there is no computable set C such that $A \subseteq C$ and $B \subseteq \overline{C}$.

Lemma 5.75. $\mathbf{Q}$ *and* $\bar{\mathbf{Q}} = \{\varphi : \mathbf{Q} \vdash \neg\varphi\}$ *are computably inseparable.*

Proof sketch. Suppose C is a computable set with $\mathbf{Q} \subseteq C$ and $\bar{\mathbf{Q}} \subseteq \overline{C}$. Define $R(x, y)$ to hold iff x codes a formula $\theta(u)$ and $\theta(\overline{y}) \in C$. Since C is computable, R is computable. We show R is a universal computable relation, contradicting the fact (established by diagonalization) that no such relation exists. If $S(y)$ is any computable relation, represented by $\theta_S(u)$ in $\mathbf{Q}$, then $S(n) \Rightarrow \mathbf{Q} \vdash \theta_S(\overline{n}) \Rightarrow \theta_S(\overline{n}) \in C \Rightarrow R({}^{\#}\theta_S(u)^{\#}, n)$, and $\neg S(n) \Rightarrow \mathbf{Q} \vdash \neg\theta_S(\overline{n}) \Rightarrow \theta_S(\overline{n}) \in \bar{\mathbf{Q}} \subseteq \overline{C} \Rightarrow \neg R({}^{\#}\theta_S(u)^{\#}, n)$. □

5.7 Theorems

We conclude the chapter with the central theorems of computability theory: Rice's theorem, the recursion (fixed-point) theorem, the unsolvability of the decision problem for first-order logic, and a collection of undecidability and incompleteness results for arithmetical theories.

5.7.1 Rice's theorem

Theorem 5.76 (Rice's Theorem)**.** *Let C be any set of partial computable functions, and let $A = \{n : \varphi_n \in C\}$. If A is computable, then either C is empty or C is the set of all partial computable functions.*

Rice's theorem says that no nontrivial *index set* is decidable. It is important to understand what the theorem does and does not say. There are certainly computable questions about programs as syntactic objects ("does this program have more than 150 symbols?"). Rice's theorem says that no non-trivial question about a program's *behavior*—what function it computes—is decidable. This includes questions like: does it halt on input 0? Does it ever halt? Does it ever output an even number?

Proof. Suppose C is neither empty nor the set of all partial computable functions, and let A be the set of indices of functions in C. We show that if A were computable, we could solve the halting problem; so A is not computable.

Without loss of generality, assume that the nowhere-defined function f is not in C (otherwise, switch C and its complement). Let g be any function in C. Define

$$h(x,y) \simeq \begin{cases} \text{undefined} & \text{if } \varphi_x(x) \uparrow \\ g(y) & \text{otherwise.} \end{cases}$$

More precisely, $h(x,y) \simeq P^2_0(g(y), \mathrm{Un}(x,x))$, which is defined and equals $g(y)$ exactly when both $\mathrm{Un}(x,x)$ and $g(y)$ are defined.

For a fixed x: if $\varphi_x(x)$ is undefined, then $h(x,y)$ is undefined for every y, so h_x acts like f; if $\varphi_x(x)$ is defined, then $h(x,y) \simeq g(y)$, so h_x acts like g.

Since h is partial computable, it equals φ_e for some e. By the *s-m-n* theorem, there is a primitive recursive s such that $\varphi_{s(e,x)}(y) = h_x(y)$. Now for each x: if $\varphi_x(x) \downarrow$, then $\varphi_{s(e,x)}$ computes $g \in C$, so $s(e,x) \in A$; if $\varphi_x(x) \uparrow$, then $\varphi_{s(e,x)}$ computes $f \notin C$, so $s(e,x) \notin A$. Hence $x \in K$ iff $s(e,x) \in A$. If A were computable, so would K—contradiction. □

Corollary 5.77. *The following sets are undecidable:*

1. $\{x : 17$ *is in the range of* $\varphi_x\}$,
2. $\{x : \varphi_x$ *is constant*$\}$,
3. $\{x : \varphi_x$ *is total*$\}$,
4. $\{x :$ *whenever* $y < y'$, $\varphi_x(y) \downarrow$, *and if* $\varphi_x(y') \downarrow$, *then* $\varphi_x(y) < \varphi_x(y')\}$.

Proof. These are all nontrivial index sets. □

5.7.2 The recursion theorem

The recursion theorem (also known as the fixed-point theorem) says that any computable transformation of programs has a fixed point. Think of it this way: given any partial computable $g(x,y)$, one can find a program e that computes $g_e(y) = g(e,y)$—a program whose behavior depends on its own code. This is the theoretical basis for self-referential constructions such as quines (programs that print their own source code).

Lemma 5.78. *The following statements are equivalent:*

1. *For every partial computable function $g(x, y)$, there is an index e such that for every y, $\varphi_e(y) \simeq g(e, y)$.*
2. *For every computable function $f(x)$, there is an index e such that for every y, $\varphi_e(y) \simeq \varphi_{f(e)}(y)$.*

Proof. (1) $\Rightarrow$ (2): Given f, define $g(x, y) \simeq \mathrm{Un}(f(x), y)$ and apply (1).

(2) $\Rightarrow$ (1): Given g, use the *s-m-n* theorem to get f such that $\varphi_{f(x)}(y) \simeq g(x, y)$ and apply (2). □

Theorem 5.79 (Recursion Theorem / Fixed-Point Theorem)**.** *For every partial computable function $g(x, y)$, there is an index e such that for every y,*

$$\varphi_e(y) \simeq g(e, y).$$

Proof. Let $\mathrm{diag}(x)$ be a computable function such that $\varphi_{\mathrm{diag}(x)}(y) \simeq \varphi_x(x, y)$. Such a function exists by the *s-m-n* theorem: define $s(x, y) \simeq \mathrm{Un}^2(x, x, y)$ and let diag satisfy $\varphi_{\mathrm{diag}(x)}(y) \simeq s(x, y)$.

Now define $l(x, y) \simeq g(\mathrm{diag}(x), y)$ and let $\ulcorner l \urcorner$ be an index for l. Set $e = \mathrm{diag}(\ulcorner l \urcorner)$. Then:

$$\begin{aligned}\varphi_e(y) &\simeq \varphi_{\mathrm{diag}(\ulcorner l \urcorner)}(y) \simeq \varphi_{\ulcorner l \urcorner}(\ulcorner l \urcorner, y) \simeq l(\ulcorner l \urcorner, y)\\ &\simeq g(\mathrm{diag}(\ulcorner l \urcorner), y) \simeq g(e, y).\end{aligned}$$ □

5.7.3 Unsolvability of the decision problem

Theorem 5.80. *The decision problem is unsolvable: there is no Turing machine D which, when started on a tape containing a sentence ψ of first-order logic as input, eventually halts and outputs* 1 *iff ψ is valid and* 0 *otherwise.*

Proof. Suppose the decision problem were solvable by a Turing machine D. We construct a Turing machine E that, given input e and w, computes the sentence $\tau(M_e, w) \to \alpha(M_e, w)$ (see Definition 5.47). The machine $E \frown D$ then computes $\tau(M_e, w) \to \alpha(M_e, w)$ and runs D on the result. By Lemma 5.50 and Lemma 5.51, $\tau(M_e, w) \to \alpha(M_e, w)$ is valid iff M_e halts on input w. So $E \frown D$ solves the halting problem, contradicting Theorem 5.40. □

Corollary 5.81. *It is undecidable whether an arbitrary sentence of first-order logic is satisfiable.*

Proof. If satisfiability were decidable by S, we could decide validity: given ψ, run S on $\neg\psi$; then ψ is valid iff $\neg\psi$ is unsatisfiable. $\square$

Theorem 5.82. *Validity of first-order sentences is semi-decidable: there is a Turing machine E which halts with output 1 iff ψ is valid, but does not halt otherwise.*

Proof. All possible derivations can be generated one after another by an effective algorithm. The machine E generates derivations and halts with output 1 when it finds one showing $\vdash \psi$. By soundness, if E halts then $\vDash \psi$. By completeness, if $\vDash \psi$ then such a derivation exists and will eventually be found. $\square$

5.7.4 Undecidability of arithmetic theories

We now connect computability to the incompleteness phenomena. The results below show that essentially any theory strong enough to represent computable functions is undecidable, and that axiomatizable extensions cannot be complete.

Theorem 5.83. *If Γ is a consistent theory that represents every decidable relation, then Γ is not decidable.*

Proof sketch. Suppose Γ were decidable. Enumerate all formulas with one free variable as $\varphi_0(x), \varphi_1(x), \ldots$ and define $D = \{n : \Gamma \vdash \neg\varphi_n(\overline{n})\}$. Since Γ is decidable, D is decidable. Since Γ represents all decidable relations, D is represented by some $\varphi_d(x)$. Then $d \in D$ leads to $\Gamma \vdash \varphi_d(\overline{d})$ (since φ_d represents D) and $\Gamma \vdash \neg\varphi_d(\overline{d})$ (by definition of D), making Γ inconsistent. $\square$

Theorem 5.84. *If Γ is axiomatizable and complete, then Γ is decidable.*

Proof sketch. If Γ is inconsistent, it is trivially decidable. Otherwise, simultaneously search for a derivation of φ and a derivation of $\neg\varphi$ from the axioms of Γ. Since Γ is complete, one search must succeed; since Γ is consistent, the other cannot. $\square$

Corollary 5.85 (Weak first incompleteness)**.** *If Γ is consistent, axiomatizable, and represents every decidable property, then Γ is not complete.*

Proof. If Γ were complete, it would be decidable (since it is axiomatizable), contradicting Theorem 5.83. □

Theorem 5.86. **Q** *is c.e. but not decidable. In fact,* **Q** *is a complete c.e. set.*

Proof sketch. **Q** is c.e. since $\mathbf{Q} = \{y : \exists x\, \mathrm{Prf}_{\mathbf{Q}}(x, y)\}$ and $\mathrm{Prf}_{\mathbf{Q}}$ is primitive recursive. To show c.e.-completeness, we reduce K to **Q**: since Kleene's predicate $T(e, x, s)$ is represented in **Q** by some φ_T, we have $x \in K$ iff $\mathbf{Q} \vdash \exists s\, \varphi_T(\overline{x}, \overline{x}, s)$. The function mapping x to (a code for) $\exists s\, \varphi_T(\overline{x}, \overline{x}, s)$ is a reduction of K to **Q**. □

Theorem 5.87. *Let* **T** *be any consistent theory that includes* **Q**. *Then* **T** *is not decidable.*

Proof sketch. If **T** were consistent and decidable, we could define a universal computable relation $R(x, y)$: "x codes a formula $\theta(u)$ and **T** proves $\theta(\overline{y})$." Since **T** extends **Q**, every computable relation $S(y)$ represented by $\theta_S(u)$ satisfies $S(n)$ iff $R(^{\#}\theta_S(u)^{\#}, n)$. But no universal computable relation exists (by diagonalization), so **T** is not decidable. □

Theorem 5.88. *Let* **T** *be any theory in the language of arithmetic that is consistent with* **Q** *(i.e.,* $\mathbf{T} \cup \mathbf{Q}$ *is consistent). Then* **T** *is undecidable.*

Proof sketch. Suppose **T** is decidable and consistent with **Q**. Let $\alpha = Q_1 \wedge \cdots \wedge Q_8$ be the conjunction of the axioms of **Q**, and define $C = \{\varphi : \mathbf{T} \vdash \alpha \rightarrow \varphi\}$. Then C is computable. If $\varphi \in \mathbf{Q}$, then $\vdash \alpha \rightarrow \varphi$, so $\varphi \in C$. If $\varphi \in \bar{\mathbf{Q}}$, then $\vdash \alpha \rightarrow \neg\varphi$; if also $\mathbf{T} \vdash \alpha \rightarrow \varphi$, then $\mathbf{T} \vdash \neg\alpha$, contradicting consistency of $\mathbf{T} \cup \mathbf{Q}$. So $\varphi \notin C$. Thus C computably separates **Q** and $\bar{\mathbf{Q}}$, contradicting Lemma 5.75. □

Theorem 5.89. *Suppose* **T** *is a theory in a language in which one can interpret the language of arithmetic, in such a way that* **T** *is consistent with the interpretation of* **Q**. *Then* **T** *is undecidable. If* **T** *proves the interpretation of the axioms of* **Q**, *then no consistent extension of* **T** *is decidable.*

Proof sketch. The proof is a small modification of the proof of Theorem 5.88: a counterexample would yield a computable separation of **Q** and $\bar{\mathbf{Q}}$ via the interpretation. □

Corollary 5.90. *There is no decidable extension of* **ZFC** *(Zermelo–Fraenkel set theory with the axiom of choice). In particular, there is no complete, consistent, axiomatizable extension of* **ZFC***.*

Corollary 5.91. *First-order logic for any language with a binary relation symbol is undecidable.*

Remark 29 (Decidability boundary). The undecidability of first-order logic extends to any language with two unary function symbols (since these can simulate a binary relation). On the other hand, first-order logic for a language with only unary relation symbols and at most one unary function symbol is decidable. Similarly, Presburger arithmetic—the set of sentences in the language 0, $'$, $+$ true in $\mathfrak{N}$—is decidable (though computationally very expensive), while the set of true sentences in the language 0, $'$, $\times$ is already undecidable.

Chapter 6

Metatheory

6.1 Soundness

The Soundness Theorem establishes that the derivability relation $\vdash$ is truth-preserving: nothing that can be derived from a set of sentences Γ goes beyond what is semantically entailed by Γ.

Theorem 6.1 (Soundness Theorem)**.** *If $\Gamma \vdash \varphi$, then $\Gamma \vDash \varphi$.*

This result is proved independently for each of the four proof systems treated in this text:

- **Axiomatic (Hilbert-style) deduction —** see §4.2.
- **Natural deduction —** see §4.3.
- **Sequent calculus —** see §4.4.
- **Tableaux —** see §4.5.

In each case, the proof proceeds by induction on the length (or tree structure) of the derivation. The base cases verify that every logical axiom (in the axiomatic system) or initial sequent / leaf assumption is logically valid or follows from the assumptions. The inductive step shows that each rule of inference—modus ponens, introduction/elimination rules, structural rules, or tableau expansion rules, depending on the system—preserves truth: if the premises of the rule are true in every structure satisfying Γ, so is the conclusion. Since every step in the derivation preserves truth, the final conclusion is entailed by Γ.

An important corollary is that every satisfiable set is consistent:

Corollary 6.2. *If Γ is satisfiable, then Γ is consistent. Equivalently, if Γ is inconsistent, then Γ is unsatisfiable.*

Proof. Suppose Γ is satisfiable and, toward a contradiction, suppose Γ is inconsistent, i.e., $\Gamma \vdash \bot$. By Soundness (6.1), $\Gamma \vDash \bot$. But $\bot$ is false in every structure, so no structure satisfies Γ—contradicting the assumption that Γ is satisfiable. □

6.2 Completeness

The Completeness Theorem, due to Gödel (1930) with an influential alternative proof by Henkin (1949), establishes the converse of soundness: every sentence that is semantically entailed by a set of sentences is derivable from it. Equivalently, every consistent set of sentences is satisfiable. We follow the Henkin proof strategy, which constructs a model out of the syntax itself.

6.2.1 Complete Consistent Sets of Sentences

Definition 6.3 (Complete set)**.** A set Γ of sentences is *complete* iff for any sentence φ, either $\varphi \in \Gamma$ or $\neg\varphi \in \Gamma$.

Example 6.4. Let $\mathfrak{M}$ be any structure for $\mathcal{L}$. The set $\Gamma = \{\varphi : \mathfrak{M} \vDash \varphi\}$ of all sentences true in $\mathfrak{M}$ is complete: for any sentence φ, either $\mathfrak{M} \vDash \varphi$ (so $\varphi \in \Gamma$) or $\mathfrak{M} \vDash \neg\varphi$ (so $\neg\varphi \in \Gamma$). It is also consistent, since $\mathfrak{M} \models \Gamma$.

Complete consistent sets are central to the completeness proof: we will show that every consistent set of sentences Γ can be extended to a complete consistent set Γ^*, from which a satisfying structure is constructed.

In what follows, we will often tacitly use the properties of reflexivity, monotonicity, and transitivity of $\vdash$ (see §4.1).

Proposition 6.5. *Suppose Γ is complete and consistent. Then:*

1. *If $\Gamma \vdash \varphi$, then $\varphi \in \Gamma$.*
2. *$\varphi \wedge \psi \in \Gamma$ iff both $\varphi \in \Gamma$ and $\psi \in \Gamma$.*
3. *$\varphi \vee \psi \in \Gamma$ iff either $\varphi \in \Gamma$ or $\psi \in \Gamma$.*
4. *$\varphi \rightarrow \psi \in \Gamma$ iff either $\varphi \notin \Gamma$ or $\psi \in \Gamma$.*

Proof. Let us suppose for all of the following that Γ is complete and consistent.

1. If $\Gamma \vdash \varphi$, then $\varphi \in \Gamma$.

   Suppose that $\Gamma \vdash \varphi$. Suppose to the contrary that $\varphi \notin \Gamma$. Since Γ is complete, $\neg\varphi \in \Gamma$. By the properties of derivability (see DEF-DED003, §4.1), Γ is inconsistent. This contradicts the assumption that Γ is consistent. Hence, it cannot be the case that $\varphi \notin \Gamma$, so $\varphi \in \Gamma$.

2. $\varphi \wedge \psi \in \Gamma$ iff both $\varphi \in \Gamma$ and $\psi \in \Gamma$:

   For the forward direction, suppose $\varphi \wedge \psi \in \Gamma$. Then by the provability properties of $\wedge$ (see §4.1), item (1), $\Gamma \vdash \varphi$ and $\Gamma \vdash \psi$. By 1, $\varphi \in \Gamma$ and $\psi \in \Gamma$, as required.

   For the reverse direction, let $\varphi \in \Gamma$ and $\psi \in \Gamma$. By the provability properties of $\wedge$, item (2), $\Gamma \vdash \varphi \wedge \psi$. By 1, $\varphi \wedge \psi \in \Gamma$.

3. First we show that if $\varphi \vee \psi \in \Gamma$, then either $\varphi \in \Gamma$ or $\psi \in \Gamma$. Suppose $\varphi \vee \psi \in \Gamma$ but $\varphi \notin \Gamma$ and $\psi \notin \Gamma$. Since Γ is complete, $\neg\varphi \in \Gamma$ and $\neg\psi \in \Gamma$. By the provability properties of $\vee$ (see §4.1), item (1), Γ is inconsistent, a contradiction. Hence, either $\varphi \in \Gamma$ or $\psi \in \Gamma$.

   For the reverse direction, suppose that $\varphi \in \Gamma$ or $\psi \in \Gamma$. By the provability properties of $\vee$, item (2), $\Gamma \vdash \varphi \vee \psi$. By 1, $\varphi \vee \psi \in \Gamma$, as required.

4. For the forward direction, suppose $\varphi \rightarrow \psi \in \Gamma$, and suppose to the contrary that $\varphi \in \Gamma$ and $\psi \notin \Gamma$. On these assumptions, $\varphi \rightarrow \psi \in \Gamma$ and $\varphi \in \Gamma$. By the provability properties of $\rightarrow$ (see §4.1), item (1), $\Gamma \vdash \psi$. But then by 1, $\psi \in \Gamma$, contradicting the assumption that $\psi \notin \Gamma$.

   For the reverse direction, first consider the case where $\varphi \notin \Gamma$. Since Γ is complete, $\neg\varphi \in \Gamma$. By the provability properties of $\rightarrow$, item (2), $\Gamma \vdash \varphi \rightarrow \psi$. Again by 1, we get that $\varphi \rightarrow \psi \in \Gamma$, as required.

   Now consider the case where $\psi \in \Gamma$. By the provability properties of $\rightarrow$, item (2) again, $\Gamma \vdash \varphi \rightarrow \psi$. By 1, $\varphi \rightarrow \psi \in \Gamma$.

□

6.2.2 Maximally Consistent Sets of Sentences

Definition 6.6 (Maximally consistent set)**.** A set Γ of sentences is *maximally consistent* iff

1. Γ is consistent, and
2. if $\Gamma \subsetneq \Gamma'$, then Γ' is inconsistent.

Example 6.7. The set $\Gamma = \{\varphi : \mathfrak{M} \vDash \varphi\}$ from Example 6.4 is maximally consistent: it is consistent (since $\mathfrak{M} \models \Gamma$), and adding any $\psi \notin \Gamma$ produces an inconsistent set, because $\psi \notin \Gamma$ means $\mathfrak{M} \vDash \neg\psi$, so $\neg\psi \in \Gamma$, and $\Gamma \cup \{\psi\} \vdash \bot$.

Every maximally consistent set is complete and consistent; it therefore has all the properties established in Proposition 6.5.

6.2.3 Henkin Expansion

In order to guarantee that the model we construct from a complete consistent set Γ makes all the quantified formulas in Γ true, we use a trick due to Leon Henkin: expand the language by infinitely many constants and add, for each formula with one free variable $\varphi(x)$, a formula of the form $\exists x\, \varphi(x) \to \varphi(c)$, where c is one of the new constants.

Proposition 6.8. *If Γ is consistent in $\mathcal{L}$ and $\mathcal{L}'$ is obtained from $\mathcal{L}$ by adding a denumerable set of new constants $d_0, d_1, \dots$, then Γ is consistent in $\mathcal{L}'$.*

Definition 6.9 (Saturated set)**.** A set Γ of formulas of a language $\mathcal{L}$ is *saturated* iff for each formula $\varphi(x) \in \mathrm{Frm}(\mathcal{L})$ with one free variable x there is a constant $c \in \mathcal{L}$ such that $\exists x\, \varphi(x) \to \varphi(c) \in \Gamma$.

Example 6.10. Let $\mathcal{L}$ contain a constant c and a one-place predicate P. If a consistent set Γ contains $\exists x\, P(x)$ and also $\exists x\, P(x) \to P(c)$, then Γ has a "witness" for its existential: the constant c tells us *which* element satisfies P. A saturated set provides such witnesses for *every* existential sentence it contains.

The following definition will be used in the proof of the next lemma.

Definition 6.11. Let $\mathcal{L}'$ be as in Proposition 6.8. Fix an enumeration $\varphi_0(x_0)$, $\varphi_1(x_1), \dots$ of all formulas $\varphi_i(x_i)$ of $\mathcal{L}'$ in which one variable (x_i) occurs free. We define the sentences θ_n by induction on n.

Let c_0 be the first constant among the d_i we added to $\mathcal{L}$ which does not occur in $\varphi_0(x_0)$. Assuming that $\theta_0, \dots, \theta_{n-1}$ have already been defined, let c_n be the first among the new constants d_i that occurs neither in $\theta_0, \dots, \theta_{n-1}$ nor in $\varphi_n(x_n)$.

Now let θ_n be the formula $\exists x_n\, \varphi_n(x_n) \to \varphi_n(c_n)$.

Lemma 6.12 (Henkin's Lemma)**.** *Every consistent set Γ can be extended to a saturated consistent set Γ'.*

Proof. Given a consistent set of sentences Γ in a language $\mathcal{L}$, expand the language by adding a denumerable set of new constants to form $\mathcal{L}'$. By Proposition 6.8, Γ is still consistent in the richer language. Further, let θ_i be as in

Definition 6.11. Let

$$\Gamma_0 = \Gamma$$
$$\Gamma_{n+1} = \Gamma_n \cup \{\theta_n\}$$

i.e., $\Gamma_{n+1} = \Gamma \cup \{\theta_0, \ldots, \theta_n\}$, and let $\Gamma' = \bigcup_n \Gamma_n$. Γ' is clearly saturated.

If Γ' were inconsistent, then for some n, Γ_n would be inconsistent. So to show that Γ' is consistent it suffices to show, by induction on n, that each set Γ_n is consistent.

The induction basis is simply the claim that $\Gamma_0 = \Gamma$ is consistent, which is the hypothesis of the lemma. For the induction step, suppose that Γ_n is consistent but $\Gamma_{n+1} = \Gamma_n \cup \{\theta_n\}$ is inconsistent. Recall that θ_n is $\exists x_n\, \varphi_n(x_n) \rightarrow \varphi_n(c_n)$, where $\varphi_n(x_n)$ is a formula of $\mathcal{L}'$ with only the variable x_n free. By the way we have chosen the c_n (see Definition 6.11), c_n does not occur in $\varphi_n(x_n)$ nor in Γ_n.

If $\Gamma_n \cup \{\theta_n\}$ is inconsistent, then $\Gamma_n \vdash \neg\theta_n$, and hence both of the following hold:

$$\Gamma_n \vdash \exists x_n\, \varphi_n(x_n) \qquad \Gamma_n \vdash \neg\varphi_n(c_n)$$

Since c_n does not occur in Γ_n or in $\varphi_n(x_n)$, the strong generalization theorem applies (see §4.1). From $\Gamma_n \vdash \neg\varphi_n(c_n)$, we obtain $\Gamma_n \vdash \forall x_n\, \neg\varphi_n(x_n)$. Thus we have that both $\Gamma_n \vdash \exists x_n\, \varphi_n(x_n)$ and $\Gamma_n \vdash \forall x_n\, \neg\varphi_n(x_n)$, so Γ_n itself is inconsistent. Contradiction: Γ_n was supposed to be consistent. Hence $\Gamma_n \cup \{\theta_n\}$ is consistent. □

We now show that complete, consistent sets which are saturated have the property that they contain an existentially quantified sentence iff they contain at least one instance, and they contain a universally quantified sentence iff they contain all instances.

Proposition 6.13. *Suppose Γ is complete, consistent, and saturated.*

1. *$\exists x\, \varphi(x) \in \Gamma$ iff $\varphi(t) \in \Gamma$ for at least one closed term t.*
2. *$\forall x\, \varphi(x) \in \Gamma$ iff $\varphi(t) \in \Gamma$ for all closed terms t.*

Proof.

1. First suppose that $\exists x\, \varphi(x) \in \Gamma$. Because Γ is saturated, $(\exists x\, \varphi(x) \rightarrow \varphi(c)) \in \Gamma$ for some constant c. By the provability properties of $\rightarrow$ (see Proposition 6.54), and Proposition 6.51, $\varphi(c) \in \Gamma$.

   For the other direction, saturation is not necessary: Suppose $\varphi(t) \in \Gamma$. Then $\Gamma \vdash \exists x\, \varphi(x)$ by the provability properties of quantifiers (see §4.1), item (1). By Proposition 6.51, $\exists x\, \varphi(x) \in \Gamma$.

2. Suppose that $\varphi(t) \in \Gamma$ for all closed terms t. By way of contradiction, assume $\forall x\, \varphi(x) \notin \Gamma$. Since Γ is complete, $\neg\forall x\, \varphi(x) \in \Gamma$. By saturation,

$(\exists x\, \neg\varphi(x) \to \neg\varphi(c)) \in \Gamma$ for some constant c. By assumption, since c is a closed term, $\varphi(c) \in \Gamma$. But this would make Γ inconsistent, since $\neg\forall x\, \varphi(x)$, $\exists x\, \neg\varphi(x) \to \neg\varphi(c)$, $\varphi(c)$ is inconsistent.

For the reverse direction, we do not need saturation: Suppose $\forall x\, \varphi(x) \in \Gamma$. Then $\Gamma \vdash \varphi(t)$ by the provability properties of quantifiers (see §4.1), item (2). We get $\varphi(t) \in \Gamma$ by Proposition 6.5.

□

6.2.4 Lindenbaum's Lemma

Lemma 6.14 (Lindenbaum's Lemma)**.** *Every consistent set Γ in a language $\mathcal{L}$ can be extended to a complete and consistent set Γ^*.*

Proof. Let Γ be consistent. Let $\varphi_0, \varphi_1, \ldots$ be an enumeration of all the sentences of $\mathcal{L}$. Define $\Gamma_0 = \Gamma$, and

$$\Gamma_{n+1} = \begin{cases} \Gamma_n \cup \{\varphi_n\} & \text{if } \Gamma_n \cup \{\varphi_n\} \text{ is consistent;} \\ \Gamma_n \cup \{\neg\varphi_n\} & \text{otherwise.} \end{cases}$$

Let $\Gamma^* = \bigcup_{n \geq 0} \Gamma_n$.

Each Γ_n is consistent: Γ_0 is consistent by definition. If $\Gamma_{n+1} = \Gamma_n \cup \{\varphi_n\}$, this is because the latter is consistent. If it is not, $\Gamma_{n+1} = \Gamma_n \cup \{\neg\varphi_n\}$. We have to verify that $\Gamma_n \cup \{\neg\varphi_n\}$ is consistent. Suppose it is not. Then *both* $\Gamma_n \cup \{\varphi_n\}$ and $\Gamma_n \cup \{\neg\varphi_n\}$ are inconsistent. This means that Γ_n would be inconsistent by the exhaustive cases property of derivability (see §4.1), contrary to the induction hypothesis.

For every n and every $i < n$, $\Gamma_i \subseteq \Gamma_n$. This follows by a simple induction on n. For $n = 0$, there are no $i < 0$, so the claim holds automatically. For the inductive step, suppose it is true for n. We show that if $i < n + 1$ then $\Gamma_i \subseteq \Gamma_{n+1}$. We have $\Gamma_{n+1} = \Gamma_n \cup \{\varphi_n\}$ or $= \Gamma_n \cup \{\neg\varphi_n\}$ by construction. So $\Gamma_n \subseteq \Gamma_{n+1}$. If $i < n + 1$, then $\Gamma_i \subseteq \Gamma_n$ by inductive hypothesis (if $i < n$) or the trivial fact that $\Gamma_n \subseteq \Gamma_n$ (if $i = n$). We get that $\Gamma_i \subseteq \Gamma_{n+1}$ by transitivity of $\subseteq$.

From this it follows that Γ^* is consistent. Here is why: Let $\Gamma' \subseteq \Gamma^*$ be finite. Each $\psi \in \Gamma'$ is also in Γ_i for some i. Let n be the largest of these. Since $\Gamma_i \subseteq \Gamma_n$ if $i \leq n$, every $\psi \in \Gamma'$ is also $\in \Gamma_n$, i.e., $\Gamma' \subseteq \Gamma_n$, and Γ_n is consistent. So, every finite subset $\Gamma' \subseteq \Gamma^*$ is consistent. By the compactness of derivability (see §4.1), Γ^* is consistent.

Every sentence of $\mathrm{Frm}(\mathcal{L})$ appears on the list used to define Γ^*. If $\varphi_n \notin \Gamma^*$, then that is because $\Gamma_n \cup \{\varphi_n\}$ was inconsistent. But then $\neg\varphi_n \in \Gamma^*$, so Γ^* is complete. □

6.2.5 Construction of a Model

We first extend Γ to a consistent, complete, and saturated set Γ^*. The term model $\mathfrak{M}(\Gamma^*)$ takes the set of closed terms of $\mathcal{L}'$ as the domain, assigns every constant to itself, and defines predicate extensions so that an atomic sentence is true in $\mathfrak{M}(\Gamma^*)$ iff it is in Γ^*.

Definition 6.15 (Term model)**.** Let Γ^* be a complete and consistent, saturated set of sentences in a language $\mathcal{L}$. The *term model* $\mathfrak{M}(\Gamma^*)$ of Γ^* is the structure defined as follows:

1. The domain $|\mathfrak{M}(\Gamma^*)|$ is the set of all closed terms of $\mathcal{L}$.
2. The interpretation of a constant c is c itself: $c^{\mathfrak{M}(\Gamma^*)} = c$.
3. The function f is assigned the function which, given as arguments the closed terms $t_1, \dots, t_n$, has as value the closed term $f(t_1, \dots, t_n)$:
$$f^{\mathfrak{M}(\Gamma^*)}(t_1, \dots, t_n) = f(t_1, \dots, t_n)$$
4. If R is an n-place predicate, then
$$\langle t_1, \dots, t_n \rangle \in R^{\mathfrak{M}(\Gamma^*)} \text{ iff } R(t_1, \dots, t_n) \in \Gamma^*.$$

Example 6.16. Suppose $\mathcal{L}$ contains a constant c, a one-place function symbol f, and a two-place predicate R. The closed terms of $\mathcal{L}$ are c, $f(c)$, $f(f(c))$, ... In the term model $\mathfrak{M}(\Gamma^*)$, the domain is $\{c, f(c), f(f(c)), \dots\}$; the interpretation of f maps each term t to the term $f(t)$; and $\langle s, t \rangle \in R^{\mathfrak{M}(\Gamma^*)}$ iff $R(s, t) \in \Gamma^*$.

Lemma 6.17. *Let $\mathfrak{M}(\Gamma^*)$ be the term model of Definition 6.15; then* $\mathrm{Val}^{\mathfrak{M}(\Gamma^*)}(t) = t$.

Proof. The proof is by induction on t, where the base case, when t is a constant, follows directly from the definition of the term model. For the induction step assume $t_1, \dots, t_n$ are closed terms such that $\mathrm{Val}^{\mathfrak{M}(\Gamma^*)}(t_i) = t_i$ and that f is an n-ary function. Then

$$\begin{aligned}
\mathrm{Val}^{\mathfrak{M}(\Gamma^*)}(f(t_1, \dots, t_n)) &= f^{\mathfrak{M}(\Gamma^*)}(\mathrm{Val}^{\mathfrak{M}(\Gamma^*)}(t_1), \dots, \mathrm{Val}^{\mathfrak{M}(\Gamma^*)}(t_n)) \\
&= f^{\mathfrak{M}(\Gamma^*)}(t_1, \dots, t_n) \\
&= f(t_1, \dots, t_n),
\end{aligned}$$

and so by induction this holds for every closed term t. □

Proposition 6.18. *Let $\mathfrak{M}(\Gamma^*)$ be the term model of Definition 6.15.*

1. $\mathfrak{M}(\Gamma^*) \vDash \exists x\, \varphi(x)$ *iff* $\mathfrak{M}(\Gamma^*) \vDash \varphi(t)$ *for at least one closed term t.*

2. $\mathfrak{M}(\Gamma^*) \vDash \forall x\, \varphi(x)$ *iff* $\mathfrak{M}(\Gamma^*) \vDash \varphi(t)$ *for all closed terms t.*

Proof.

1. By the definition of satisfaction (see DEF-SEM002, §3.4), $\mathfrak{M}(\Gamma^*) \vDash \exists x\, \varphi(x)$ iff for at least one variable assignment s, $\mathfrak{M}(\Gamma^*), s \vDash \varphi(x)$. As $|\mathfrak{M}(\Gamma^*)|$ consists of the closed terms of $\mathcal{L}$, this is the case iff there is at least one closed term t such that $s(x) = t$ and $\mathfrak{M}(\Gamma^*), s \vDash \varphi(x)$. By the Extension Lemma (see §3.4), $\mathfrak{M}(\Gamma^*), s \vDash \varphi(x)$ iff $\mathfrak{M}(\Gamma^*), s \vDash \varphi(t)$, where $s(x) = t$. By the Sentence Satisfaction Lemma (see §3.4), $\mathfrak{M}(\Gamma^*), s \vDash \varphi(t)$ iff $\mathfrak{M}(\Gamma^*) \vDash \varphi(t)$, since $\varphi(t)$ is a sentence.

2. By the definition of satisfaction, $\mathfrak{M}(\Gamma^*) \vDash \forall x\, \varphi(x)$ iff for every variable assignment s, $\mathfrak{M}(\Gamma^*), s \vDash \varphi(x)$. Recall that $|\mathfrak{M}(\Gamma^*)|$ consists of the closed terms of $\mathcal{L}$, so for every closed term t, $s(x) = t$ is such a variable assignment, and for any variable assignment, $s(x)$ is some closed term t. By the Extension Lemma, $\mathfrak{M}(\Gamma^*), s \vDash \varphi(x)$ iff $\mathfrak{M}(\Gamma^*), s \vDash \varphi(t)$, where $s(x) = t$. By the Sentence Satisfaction Lemma, $\mathfrak{M}(\Gamma^*), s \vDash \varphi(t)$ iff $\mathfrak{M}(\Gamma^*) \vDash \varphi(t)$, since $\varphi(t)$ is a sentence.

□

Lemma 6.19 (Truth Lemma)**.** *Suppose φ does not contain $=$. Then $\mathfrak{M}(\Gamma^*) \vDash \varphi$ iff $\varphi \in \Gamma^*$.*

Proof. We prove both directions simultaneously, and by induction on φ.

1. $\varphi \equiv \bot$: $\mathfrak{M}(\Gamma^*) \nvDash \bot$ by definition of satisfaction. On the other hand, $\bot \notin \Gamma^*$ since Γ^* is consistent.

2. $\varphi \equiv \top$: $\mathfrak{M}(\Gamma^*) \vDash \top$ by definition of satisfaction. On the other hand, $\top \in \Gamma^*$ since Γ^* is consistent and complete, and $\Gamma^* \vdash \top$.

3. $\varphi \equiv R(t_1, \dots, t_n)$: $\mathfrak{M}(\Gamma^*) \vDash R(t_1, \dots, t_n)$ iff $\langle t_1, \dots, t_n \rangle \in R^{\mathfrak{M}(\Gamma^*)}$ (by the definition of satisfaction) iff $R(t_1, \dots, t_n) \in \Gamma^*$ (by the construction of $\mathfrak{M}(\Gamma^*)$).

4. $\varphi \equiv \neg\psi$: $\mathfrak{M}(\Gamma^*) \vDash \neg\psi$ iff $\mathfrak{M}(\Gamma^*) \nvDash \psi$ (by definition of satisfaction). By induction hypothesis, $\mathfrak{M}(\Gamma^*) \nvDash \psi$ iff $\psi \notin \Gamma^*$. Since Γ^* is consistent and complete, $\psi \notin \Gamma^*$ iff $\neg\psi \in \Gamma^*$.

5. $\varphi \equiv \psi \wedge \chi$: $\mathfrak{M}(\Gamma^*) \vDash \psi \wedge \chi$ iff we have both $\mathfrak{M}(\Gamma^*) \vDash \psi$ and $\mathfrak{M}(\Gamma^*) \vDash \chi$ (by definition of satisfaction) iff both $\psi \in \Gamma^*$ and $\chi \in \Gamma^*$ (by the induction hypothesis). By Proposition 6.52, this is the case iff $(\psi \wedge \chi) \in \Gamma^*$.

6. $\varphi \equiv \psi \lor \chi$: $\mathfrak{M}(\Gamma^*) \vDash \psi \lor \chi$ iff $\mathfrak{M}(\Gamma^*) \vDash \psi$ or $\mathfrak{M}(\Gamma^*) \vDash \chi$ (by definition of satisfaction) iff $\psi \in \Gamma^*$ or $\chi \in \Gamma^*$ (by induction hypothesis). This is the case iff $(\psi \lor \chi) \in \Gamma^*$ (by Proposition 6.53).

7. $\varphi \equiv \psi \rightarrow \chi$: $\mathfrak{M}(\Gamma^*) \vDash \psi \rightarrow \chi$ iff $\mathfrak{M}(\Gamma^*) \nvDash \psi$ or $\mathfrak{M}(\Gamma^*) \vDash \chi$ (by definition of satisfaction) iff $\psi \notin \Gamma^*$ or $\chi \in \Gamma^*$ (by induction hypothesis). This is the case iff $(\psi \rightarrow \chi) \in \Gamma^*$ (by Proposition 6.54).

8. $\varphi \equiv \forall x\, \psi(x)$: $\mathfrak{M}(\Gamma^*) \vDash \forall x\, \psi(x)$ iff $\mathfrak{M}(\Gamma^*) \vDash \psi(t)$ for all terms t (Proposition 6.18). By induction hypothesis, this is the case iff $\psi(t) \in \Gamma^*$ for all terms t; by Proposition 6.13, this in turn is the case iff $\forall x\, \varphi(x) \in \Gamma^*$.

9. $\varphi \equiv \exists x\, \psi(x)$: $\mathfrak{M}(\Gamma^*) \vDash \exists x\, \psi(x)$ iff $\mathfrak{M}(\Gamma^*) \vDash \psi(t)$ for at least one term t (Proposition 6.18). By induction hypothesis, this is the case iff $\psi(t) \in \Gamma^*$ for at least one term t. By Proposition 6.13, this in turn is the case iff $\exists x\, \psi(x) \in \Gamma^*$.

□

6.2.6 Identity

The term model constructed above suffices for sets Γ that do not contain the identity predicate $=$. When Γ^* contains a sentence $t = t'$ with t and t' distinct terms, the term model falsifies it (since $\mathrm{Val}^{\mathfrak{M}(\Gamma^*)}(t) = t \neq t'$). We fix this by quotienting the term model by provable equality.

Definition 6.20. Let Γ^* be a consistent and complete set of sentences in $\mathcal{L}$. We define the relation $\approx$ on the set of closed terms of $\mathcal{L}$ by

$$t \approx t' \quad \text{iff} \quad t = t' \in \Gamma^*$$

Proposition 6.21. *The relation $\approx$ has the following properties:*

1. *$\approx$ is reflexive.*
2. *$\approx$ is symmetric.*
3. *$\approx$ is transitive.*
4. *If $t \approx t'$, f is a function, and $t_1, \dots, t_{i-1}, t_{i+1}, \dots, t_n$ are closed terms, then*

$$f(t_1, \dots, t_{i-1}, t, t_{i+1}, \dots, t_n) \approx f(t_1, \dots, t_{i-1}, t', t_{i+1}, \dots, t_n).$$

5. *If $t \approx t'$, R is a predicate, and $t_1, \dots, t_{i-1}, t_{i+1}, \dots, t_n$ are closed terms, then*

$$R(t_1, \dots, t_{i-1}, t, t_{i+1}, \dots, t_n) \in \Gamma^* \text{ iff } R(t_1, \dots, t_{i-1}, t', t_{i+1}, \dots, t_n) \in \Gamma^*.$$

Proof. Since Γ^* is consistent and complete, $t = t' \in \Gamma^*$ iff $\Gamma^* \vdash t = t'$. Thus it is enough to show the following:

1. $\Gamma^* \vdash t = t$ for all closed terms t.
2. If $\Gamma^* \vdash t = t'$ then $\Gamma^* \vdash t' = t$.
3. If $\Gamma^* \vdash t = t'$ and $\Gamma^* \vdash t' = t''$, then $\Gamma^* \vdash t = t''$.
4. If $\Gamma^* \vdash t = t'$, then

$$\Gamma^* \vdash f(t_1, \dots, t_{i-1}, t, t_{i+1}, \dots, t_n) = f(t_1, \dots, t_{i-1}, t', t_{i+1}, \dots, t_n)$$

for every n-place function f and closed terms $t_1, \dots, t_{i-1}, t_{i+1}, \dots, t_n$.

5. If $\Gamma^* \vdash t = t'$ and $\Gamma^* \vdash R(t_1, \dots, t_{i-1}, t, t_{i+1}, \dots, t_n)$, then $\Gamma^* \vdash R(t_1, \dots, t_{i-1}, t', t_{i+1}, \dots, t_n)$ for every n-place predicate R and closed terms $t_1, \dots, t_{i-1}, t_{i+1}, \dots, t_n$.

□

Definition 6.22. Suppose Γ^* is a consistent and complete set in a language $\mathcal{L}$, t is a closed term, and $\approx$ as in the previous definition. Then:

$$[t]_\approx = \{t' : t' \in \mathrm{Trm}(\mathcal{L}), t \approx t'\}$$

and $\mathrm{Trm}(\mathcal{L})/_\approx = \{[t]_\approx : t \in \mathrm{Trm}(\mathcal{L})\}$.

Definition 6.23 (Factored term model)**.** Let $\mathfrak{M} = \mathfrak{M}(\Gamma^*)$ be the term model for Γ^* from Definition 6.15. Then $\mathfrak{M}/_\approx$ is the following structure:

1. $|\mathfrak{M}/_\approx| = \mathrm{Trm}(\mathcal{L})/_\approx$.
2. $c^{\mathfrak{M}/_\approx} = [c]_\approx$
3. $f^{\mathfrak{M}/_\approx}([t_1]_\approx, \dots, [t_n]_\approx) = [f(t_1, \dots, t_n)]_\approx$
4. $\langle [t_1]_\approx, \dots, [t_n]_\approx \rangle \in R^{\mathfrak{M}/_\approx}$ iff $\mathfrak{M} \vDash R(t_1, \dots, t_n)$, i.e., iff $R(t_1, \dots, t_n) \in \Gamma^*$.

The definitions of $f^{\mathfrak{M}/_\approx}$ and $R^{\mathfrak{M}/_\approx}$ refer to elements of $\mathrm{Trm}(\mathcal{L})/_\approx$ via representatives $t \in [t]_\approx$. Proposition 6.21 guarantees that these definitions do not depend on the choice of representatives.

Proposition 6.24. *$\mathfrak{M}/_\approx$ is well defined, i.e., if $t_1, \dots, t_n, t'_1, \dots, t'_n$ are closed terms, and $t_i \approx t'_i$ then*

1. *$[f(t_1, \dots, t_n)]_\approx = [f(t'_1, \dots, t'_n)]_\approx$, i.e.,*

$$f(t_1, \dots, t_n) \approx f(t'_1, \dots, t'_n)$$

and

2. *$\mathfrak{M} \vDash R(t_1, \dots, t_n)$ iff $\mathfrak{M} \vDash R(t'_1, \dots, t'_n)$, i.e.,*

$$R(t_1, \dots, t_n) \in \Gamma^* \text{ iff } R(t'_1, \dots, t'_n) \in \Gamma^*.$$

Proof. Follows from Proposition 6.21 by induction on n. □

Lemma 6.25. *Let $\mathfrak{M} = \mathfrak{M}(\Gamma^*)$; then $\mathrm{Val}^{\mathfrak{M}/_\approx}(t) = [t]_\approx$.*

Proof. The proof is similar to that of Lemma 6.17. □

Lemma 6.26 (Truth Lemma, with identity)**.** *$\mathfrak{M}/_\approx \vDash \varphi$ iff $\varphi \in \Gamma^*$ for all sentences φ.*

Proof. By induction on φ, just as in the proof of Lemma 6.19. The only case that needs additional attention is when $\varphi \equiv t = t'$.

$$\begin{aligned} \mathfrak{M}/_\approx \vDash t = t' & \text{ iff } [t]_\approx = [t']_\approx \text{ (by definition of } \mathfrak{M}/_\approx) \\ & \text{ iff } t \approx t' \text{ (by definition of } [t]_\approx) \\ & \text{ iff } t = t' \in \Gamma^* \text{ (by definition of } \approx). \end{aligned}$$

□

6.2.7 The Completeness Theorem

Let us combine our results: we arrive at the completeness theorem.

Theorem 6.27 (Completeness Theorem)**.** *Let Γ be a set of sentences. If Γ is consistent, it is satisfiable.*

Proof. Suppose Γ is consistent. By Lemma 6.12, there is a saturated consistent set $\Gamma' \supseteq \Gamma$. By Lemma 6.14 (Lindenbaum's Lemma), there is a $\Gamma^* \supseteq \Gamma'$ which is consistent and complete. Since $\Gamma' \subseteq \Gamma^*$, for each formula $\varphi(x)$, Γ^* contains a sentence of the form $\exists x\, \varphi(x) \rightarrow \varphi(c)$ and so Γ^* is saturated. If Γ does not contain $=$, then by Lemma 6.19 (Truth Lemma), $\mathfrak{M}(\Gamma^*) \vDash \varphi$ iff $\varphi \in \Gamma^*$. From this it follows in particular that for all $\varphi \in \Gamma$, $\mathfrak{M}(\Gamma^*) \vDash \varphi$, so Γ is satisfiable. If Γ does contain $=$, then by Lemma 6.26, for all sentences φ, $\mathfrak{M}/_{\approx} \vDash \varphi$ iff $\varphi \in \Gamma^*$. In particular, $\mathfrak{M}/_{\approx} \vDash \varphi$ for all $\varphi \in \Gamma$, so Γ is satisfiable. □

Corollary 6.28 (Completeness Theorem, Second Version)**.** *For all Γ and sentences φ: if $\Gamma \vDash \varphi$ then $\Gamma \vdash \varphi$.*

Proof. Note that the Γ's in Corollary 6.28 and Theorem 6.27 are universally quantified. To make sure we do not confuse ourselves, let us restate Theorem 6.27 using a different variable: for any set of sentences Δ, if Δ is consistent, it is satisfiable. By contraposition, if Δ is not satisfiable, then Δ is inconsistent. We will use this to prove the corollary.

Suppose that $\Gamma \vDash \varphi$. Then $\Gamma \cup \{\neg\varphi\}$ is unsatisfiable by the entailment–unsatisfiability equivalence (see DEF-SEM002, §3.4). Taking $\Gamma \cup \{\neg\varphi\}$ as our Δ, the previous version of Theorem 6.27 gives us that $\Gamma \cup \{\neg\varphi\}$ is inconsistent. By the derivability from inconsistency property (see §4.1), $\Gamma \vdash \varphi$. □

6.3 Compactness

Definition 6.29 (Finitely satisfiable)**.** A set Γ of formulas is *finitely satisfiable* iff every finite $\Gamma_0 \subseteq \Gamma$ is satisfiable.

Theorem 6.30 (Compactness Theorem)**.** *The following hold for any sentences Γ and φ:*

1. *$\Gamma \vDash \varphi$ iff there is a finite $\Gamma_0 \subseteq \Gamma$ such that $\Gamma_0 \vDash \varphi$.*
2. *Γ is satisfiable iff it is finitely satisfiable.*

Proof. We prove (2). If Γ is satisfiable, then there is a structure $\mathfrak{M}$ such that $\mathfrak{M} \vDash \varphi$ for all $\varphi \in \Gamma$. Of course, this $\mathfrak{M}$ also satisfies every finite subset of Γ, so Γ is finitely satisfiable.

Now suppose that Γ is finitely satisfiable. Then every finite subset $\Gamma_0 \subseteq \Gamma$ is satisfiable. By soundness (Corollary 6.2, from §6.1), every finite subset is consistent. Then Γ itself must be consistent by the compactness of derivability (see §4.1). By the Completeness Theorem (6.27), since Γ is consistent, it is satisfiable. □

Example 6.31 (De Bruijn–Erdős)**.** Every graph in which every finite subgraph is k-colorable is itself k-colorable. To see this, let $\mathcal{L}$ have a constant for each vertex and k unary predicates $C_1, \ldots, C_k$. For each vertex v, include $C_1(v) \vee \cdots \vee C_k(v)$; for each edge $\{u, v\}$ and color i, include $\neg(C_i(u) \wedge C_i(v))$. Every finite subset of this set mentions finitely many vertices, hence is satisfiable by hypothesis. By the Compactness Theorem, the whole set is satisfiable—giving a k-coloring.

6.4 The Löwenheim–Skolem Theorem

First-order logic cannot express that the size of a structure is non-enumerable: any sentence or set of sentences satisfied in all non-enumerable structures is also satisfied in some enumerable structure.

Theorem 6.32 (Downward Löwenheim–Skolem)**.** *If Γ is consistent then it has an enumerable model, i.e., it is satisfiable in a structure whose domain is either finite or denumerable.*

Proof. If Γ is consistent, the structure $\mathfrak{M}$ delivered by the proof of the Completeness Theorem (6.27) has a domain $|\mathfrak{M}|$ that is no larger than the set of the terms of the language $\mathcal{L}$. So $\mathfrak{M}$ is at most denumerable. □

Theorem 6.33 (Löwenheim–Skolem without identity)**.** *If Γ is a consistent set of sentences in the language of first-order logic without identity, then it has a denumerable model, i.e., it is satisfiable in a structure whose domain is infinite and enumerable.*

Proof. If Γ is consistent and contains no sentences in which identity appears, then the structure $\mathfrak{M}$ delivered by the proof of the Completeness Theorem has a domain $|\mathfrak{M}|$ identical to the set of terms of the language $\mathcal{L}'$. So $\mathfrak{M}$ is denumerable, since $\mathrm{Trm}(\mathcal{L}')$ is. □

Remark 30 (Skolem's paradox)*.* If ZFC is consistent, the Löwenheim–Skolem theorem gives it an enumerable model $\mathfrak{M}$. Yet ZFC proves "there exist uncountable sets." The resolution: $\mathfrak{M}$ thinks some set is uncountable because it lacks a witnessing bijection—the bijection exists outside $\mathfrak{M}$ but not inside it. "Uncountable" is not absolute but depends on the ambient model (Skolem, 1922).

6.5 The First Incompleteness Theorem

The First Incompleteness Theorem establishes an inherent limitation of sufficiently strong, axiomatizable theories: no such theory can be both consistent and complete. The proof rests on the Fixed-Point Lemma, Σ_1-completeness, and a careful analysis of the provability predicate.

6.5.1 The Fixed-Point Lemma

The fixed-point lemma says that for any formula $\psi(x)$, there is a sentence φ such that $\mathbf{T} \vdash \varphi \leftrightarrow \psi(\ulcorner\varphi\urcorner)$, provided $\mathbf{T}$ extends $\mathbf{Q}$. In the case of the liar sentence, we would want φ to be equivalent (provably in $\mathbf{T}$) to "$\ulcorner\varphi\urcorner$ is false," i.e., the statement that ${}^{\#}\varphi^{\#}$ is the Gödel number of a false sentence. To understand the idea of the proof, it will be useful to compare it with Quine's informal gloss of φ as, "'yields a falsehood when preceded by its own quotation' yields a falsehood when preceded by its own quotation." The operation of taking an expression, and then forming a sentence by preceding this expression by its own quotation may be called *diagonalizing* the expression, and the result its diagonalization. So, the diagonalization of 'yields a falsehood when preceded by its own quotation' is "'yields a falsehood when preceded by its own quotation' yields a falsehood when preceded by its own quotation." Now note that Quine's liar sentence is not the diagonalization of 'yields a falsehood' but of 'yields a falsehood when preceded by its own quotation.' So the property being diagonalized to yield the liar sentence itself involves diagonalization!

In the language of arithmetic, we form quotations of a formula with one free variable by computing its Gödel numbers and then substituting the standard numeral for that Gödel number into the free variable. The diagonalization of $\alpha(x)$ is $\alpha(\overline{n})$, where $n = {}^{\#}\alpha(x)^{\#}$. (From now on, let us abbreviate ${}^{\#}\alpha(x)^{\#}$ as $\ulcorner\alpha(x)\urcorner$.) So if $\psi(x)$ is "is a falsehood," then "yields a falsehood if preceded by its own quotation," would be "yields a falsehood when applied to the Gödel number of its diagonalization." If we had a symbol diag for the function $\mathrm{diag}(n)$ which computes the Gödel number of the diagonalization of the formula with Gödel number n, we could write $\alpha(x)$ as $\psi(\mathsf{diag}(x))$. And Quine's version of the liar sentence would then be the diagonalization of it, i.e., $\alpha(\ulcorner\alpha(x)\urcorner)$ or $\psi(\mathsf{diag}(\ulcorner\psi(\mathsf{diag}(x))\urcorner))$. Of course, $\psi(x)$ could now be any other property, and the same construction would work. For the incompleteness theorem, we take $\psi(x)$ to be "x is not derivable in $\mathbf{T}$." Then $\alpha(x)$ would be "yields a sentence not derivable in $\mathbf{T}$ when applied to the Gödel number of its diagonalization."

To formalize this in $\mathbf{T}$, we have to find a way to formalize diag. The function $\mathrm{diag}(n)$ is computable, in fact, it is primitive recursive: if n is the Gödel number of a formula $\alpha(x)$, $\mathrm{diag}(n)$ returns the Gödel number of $\alpha(\ulcorner\alpha(x)\urcorner)$. (Recall, $\ulcorner\alpha(x)\urcorner$ is the standard numeral of the Gödel number of $\alpha(x)$, i.e., ${}^{\#}\alpha(x)^{\#}$). If diag were a function symbol in $\mathbf{T}$ representing the function diag, we could take φ to be the formula $\psi(\mathsf{diag}(\ulcorner\psi(\mathsf{diag}(x))\urcorner))$. Notice that

$$\begin{aligned}\mathrm{diag}({}^{\#}\psi(\mathsf{diag}(x))^{\#}) &= {}^{\#}\psi(\mathsf{diag}(\ulcorner\psi(\mathsf{diag}(x))\urcorner))^{\#}\\ &= {}^{\#}\varphi^{\#}.\end{aligned}$$

Assuming $\mathbf{T}$ can derive

$$\mathsf{diag}(\ulcorner\psi(\mathsf{diag}(x))\urcorner) = \ulcorner\varphi\urcorner,$$

it can derive $\psi(\mathit{diag}(\ulcorner\psi(\mathit{diag}(x))\urcorner)) \leftrightarrow \psi(\ulcorner\varphi\urcorner)$. But the left hand side is, by definition, φ.

Of course, diag will in general not be a function symbol of **T**, and certainly is not one of **Q**. But, since diag is computable, it is *representable* in **Q** by some formula $\theta_{\text{diag}}(x, y)$. So instead of writing $\psi(\mathit{diag}(x))$ we can write $\exists y\,(\theta_{\text{diag}}(x, y) \land \psi(y))$. Otherwise, the proof sketched above goes through, and in fact, it goes through already in **Q**.

Lemma 6.34 (Fixed-Point Lemma)**.** *Let $\psi(x)$ be any formula with one free variable x. Then there is a sentence φ such that $\mathbf{Q} \vdash \varphi \leftrightarrow \psi(\ulcorner\varphi\urcorner)$.*

Proof. Given $\psi(x)$, let $\alpha(x)$ be the formula $\exists y\,(\theta_{\text{diag}}(x, y) \land \psi(y))$ and let φ be its diagonalization, i.e., the formula $\alpha(\ulcorner\alpha(x)\urcorner)$.

Since θ_{diag} represents diag, and $\text{diag}({}^{\#}\alpha(x)^{\#}) = {}^{\#}\varphi^{\#}$, **Q** can derive

$$\theta_{\text{diag}}(\ulcorner\alpha(x)\urcorner, \ulcorner\varphi\urcorner) \tag{6.1}$$

$$\forall y\,(\theta_{\text{diag}}(\ulcorner\alpha(x)\urcorner, y) \to y = \ulcorner\varphi\urcorner). \tag{6.2}$$

Now we show that $\mathbf{Q} \vdash \varphi \leftrightarrow \psi(\ulcorner\varphi\urcorner)$. We argue informally, using just logic and facts derivable in **Q**.

First, suppose φ, i.e., $\alpha(\ulcorner\alpha(x)\urcorner)$. Going back to the definition of $\alpha(x)$, we see that $\alpha(\ulcorner\alpha(x)\urcorner)$ just is

$$\exists y\,(\theta_{\text{diag}}(\ulcorner\alpha(x)\urcorner, y) \land \psi(y)).$$

Consider such a y. Since $\theta_{\text{diag}}(\ulcorner\alpha(x)\urcorner, y)$, by (6.2), $y = \ulcorner\varphi\urcorner$. So, from $\psi(y)$ we have $\psi(\ulcorner\varphi\urcorner)$.

Now suppose $\psi(\ulcorner\varphi\urcorner)$. By (6.1), we have

$$\theta_{\text{diag}}(\ulcorner\alpha(x)\urcorner, \ulcorner\varphi\urcorner) \land \psi(\ulcorner\varphi\urcorner).$$

It follows that

$$\exists y\,(\theta_{\text{diag}}(\ulcorner\alpha(x)\urcorner, y) \land \psi(y)).$$

But that is just $\alpha(\ulcorner\alpha(x)\urcorner)$, i.e., φ. □

6.5.2 Σ_1-Completeness

Despite the incompleteness of **Q** and its consistent, axiomatizable extensions, **Q** does prove many basic facts about numerals. In fact, this can be extended quite considerably. To understand the scope of what can be proved in **Q**, we introduce the notions of Δ_0, Σ_1, and Π_1 formulas. Roughly speaking, a Σ_1 formula is one of the form $\exists x\,\psi(x)$, where ψ is constructed using only propositional connectives and bounded quantifiers. We shall show that if φ is a Σ_1 sentence which is true in $\mathfrak{N}$ (the standard model of arithmetic; see §3.5), then $\mathbf{Q} \vdash \varphi$.

Definition 6.35 (Bounded quantifiers)**.** A *bounded existential formula* is one of the form $\exists x\,(x < t \wedge \varphi(x))$ where t is any term, which we conventionally write as $(\exists x < t)\ \varphi(x)$. A *bounded universal formula* is one of the form $\forall x\,(x < t \rightarrow \varphi(x))$ where t is any term, which we conventionally write as $(\forall x < t)\ \varphi(x)$.

Example 6.36. The formula $(\forall x < \overline{3})\ P(x)$ abbreviates $\forall x\,(x < \overline{3} \rightarrow P(x))$. In the standard model $\mathfrak{N}$, this says that P holds of 0, 1, and 2. It is a Δ_0 formula since all quantifiers are bounded.

Remark 31 (Δ_0, Σ_1, Π_1 formulas). Recall (DEF-SYN009/010, §2.5): a formula is Δ_0 if it is built from atomic formulas using only propositional connectives and bounded quantification; Σ_1 if it has the form $\exists x\,\psi(x)$ with ψ being Δ_0; and Π_1 if it has the form $\forall x\,\psi(x)$ with ψ being Δ_0.

Lemma 6.37. *Suppose t is a closed term such that* $\mathrm{Val}^{\mathfrak{N}}(t) = n$. *Then* $\mathbf{Q} \vdash t = \overline{n}$.

Proof. We prove this by induction on the complexity of t. For the base case, $\mathrm{Val}^{\mathfrak{N}}(\mathrm{o}) = 0$, and $\mathbf{Q} \vdash \mathrm{o} = \overline{0}$ since $\overline{0} \equiv \mathrm{o}$. For the inductive case, let t_1 and t_2 be terms such that $\mathrm{Val}^{\mathfrak{N}}(t_1) = n_1$, $\mathrm{Val}^{\mathfrak{N}}(t_2) = n_2$, $\mathbf{Q} \vdash t_1 = \overline{n}_1$, and $\mathbf{Q} \vdash t_2 = \overline{n}_2$.

Then $\mathrm{Val}^{\mathfrak{N}}((t_1')) = n_1 + 1$, and we have that $\mathbf{Q} \vdash t_1' = \overline{n}_1{}'$ by the first-order rules for identity applied to the induction hypothesis and the formula $\overline{n_1}' = \overline{n_1}'$, so we have $\mathbf{Q} \vdash t_1' = \overline{n_1 + 1}$ by the definition of numerals.

For sums we have

$$\mathrm{Val}^{\mathfrak{N}}((t_1 + t_2)) = \mathrm{Val}^{\mathfrak{N}}(t_1) + \mathrm{Val}^{\mathfrak{N}}(t_2) = n_1 + n_2.$$

By the induction hypothesis and the rules for identity, $\mathbf{Q} \vdash t_1 + t_2 = \overline{n_1} + t_2$, and then $\mathbf{Q} \vdash t_1 + t_2 = \overline{n_1} + \overline{n_2}$ by a second application of the rules for identity. By the fact that $\mathbf{Q}$ proves the standard addition identities (see DEF-CMP009, Representability, §5.5), $\mathbf{Q} \vdash \overline{n_1} + \overline{n_2} = \overline{n_1 + n_2}$, so $\mathbf{Q} \vdash t_1 + t_2 = \overline{n_1 + n_2}$.

Similar reasoning also works for $\times$, using the corresponding multiplication identities. Since this exhausts the closed terms of arithmetic, we have that $\mathbf{Q} \vdash t = \overline{n}$ for all closed terms t such that $\mathrm{Val}^{\mathfrak{N}}(t) = n$. □

Lemma 6.38. *Suppose t_1 and t_2 are closed terms. Then*

1. *If* $\mathrm{Val}^{\mathfrak{N}}(t_1) = \mathrm{Val}^{\mathfrak{N}}(t_2)$, *then* $\mathbf{Q} \vdash t_1 = t_2$.
2. *If* $\mathrm{Val}^{\mathfrak{N}}(t_1) \neq \mathrm{Val}^{\mathfrak{N}}(t_2)$, *then* $\mathbf{Q} \vdash t_1 \neq t_2$.
3. *If* $\mathrm{Val}^{\mathfrak{N}}(t_1) < \mathrm{Val}^{\mathfrak{N}}(t_2)$, *then* $\mathbf{Q} \vdash t_1 < t_2$.

4. *If* $\mathrm{Val}^{\mathfrak{N}}(t_2) \leq \mathrm{Val}^{\mathfrak{N}}(t_1)$, *then* $\mathbf{Q} \vdash \neg(t_1 < t_2)$.

Proof. Given terms t_1 and t_2, we fix $n = \mathrm{Val}^{\mathfrak{N}}(t_1)$ and $m = \mathrm{Val}^{\mathfrak{N}}(t_2)$.

Suppose $\varphi \equiv t_1 = t_2$. By Lemma 6.37, $\mathbf{Q} \vdash t_1 = \overline{n}$ and $\mathbf{Q} \vdash t_2 = \overline{n}$. If $n = m$, then $\mathbf{Q} \vdash \overline{n} = \overline{m}$ and hence $\mathbf{Q} \vdash t_1 = t_2$ by the transitivity of identity. If $n \neq m$ then $\mathbf{Q} \vdash \overline{n} \neq \overline{m}$, and by the transitivity of identity again, $\mathbf{Q} \vdash t_1 \neq t_2$.

Now let $\varphi \equiv t_1 < t_2$. For both cases, we rely on axiom Q_8, which states that $x < y \leftrightarrow \exists z\, z' + x = y$ for all x, y.

Suppose $\mathfrak{N} \vDash t_1 < t_2$. Then there exists some $k \in \mathbb{N}$ such that $n + k + 1 = m$. By Lemma 6.37, $\mathbf{Q} \vdash t_1 = \overline{n}$ and $\mathbf{Q} \vdash t_2 = \overline{m}$, and by the first part of this lemma, $\mathbf{Q} \vdash \overline{n} + \overline{k}' = \overline{m}$. By the transitivity of identity it follows that $\mathbf{Q} \vdash \overline{k}' + t_1 = t_2$, so $\mathbf{Q} \vdash \exists z\, z' + t_1 = t_2$. By the right-to-left direction of Q_8, $\mathbf{Q} \vdash t_1 < t_2$.

Suppose instead that $\mathfrak{N} \nvDash t_1 < t_2$, i.e., $m \leq n$. We work in $\mathbf{Q}$ and assume that $t_1 < t_2$. By the left-to-right direction of Q_8, there is some z such that $z' + t_1 = t_2$. Since $\mathbf{Q} \vdash t_1 = \overline{n}$ and $\mathbf{Q} \vdash t_2 = \overline{m}$, $z' + \overline{n} = \overline{m}$. By an external induction on m using Q_5, $z' + \overline{n - m} = \mathsf{o}$. If $m = n$ then $z' \neq \mathsf{o}$, giving a contradiction via Q_3. If $m < n$ then $(z' + \overline{n - m - 1})' = \mathsf{o}$ by Q_5 again, giving a contradiction via Q_3. So $\mathbf{Q} \vdash \neg(t_1 < t_2)$. □

Lemma 6.39. *Suppose φ is a formula, t a closed term, and $k = \mathrm{Val}^{\mathfrak{N}}(t)$. Then*

1. $\mathbf{Q} \vdash (\forall x < t)\ \varphi(x)$ *iff* $\mathbf{Q} \vdash \varphi(\overline{0}) \land \cdots \land \varphi(\overline{k-1})$.
2. $\mathbf{Q} \vdash (\exists x < t)\ \varphi(x)$ *iff* $\mathbf{Q} \vdash \varphi(\overline{0}) \lor \cdots \lor \varphi(\overline{k-1})$.

Proof. We prove the case for the bounded universal quantifier. If $\mathrm{Val}^{\mathfrak{N}}(t) = 0$ then the left-hand side of the equivalence is provable in $\mathbf{Q}$, because there is no $x < \overline{0}$ by properties of $<$ in $\mathbf{Q}$. Similarly, we can take an empty disjunction to be simply $\top$, which is also provable in $\mathbf{Q}$. We therefore suppose that $\mathrm{Val}^{\mathfrak{N}}(t) = k + 1$ for some natural number k. By Lemma 6.37 we can assume that we are working with a formula of the form $(\forall x < \overline{k+1})\ \varphi(x)$.

Suppose that $\mathbf{Q} \vdash (\forall x < \overline{k+1})\ \varphi(x)$, and let $n \leq k$. Since $\mathbf{Q} \vdash \overline{n} < \overline{k+1}$ by Lemma 6.38, it follows by logic that $\mathbf{Q} \vdash \varphi(\overline{n})$. Applying this fact $k+1$ times for each $n \leq k$, we get that $\mathbf{Q} \vdash \varphi(\overline{0}) \land \cdots \land \varphi(\overline{k})$ as desired.

For the other direction, suppose that $\mathbf{Q} \vdash \varphi(\overline{0}) \land \cdots \land \varphi(\overline{k})$. Working in $\mathbf{Q}$, suppose that $x < \overline{k+1}$. By properties of $<$ in $\mathbf{Q}$ we have that $x = \overline{0} \lor \cdots \lor x = \overline{k}$, so by logic it follows that $\varphi(x)$, and hence the universal claim $(\forall x < \overline{k+1})\ \varphi(x)$ follows.

The proof of the equivalence for bounded existentially quantified formulas is similar. □

Lemma 6.40. *If φ is a Δ_0 sentence which is true in $\mathfrak{N}$, then $\mathbf{Q} \vdash \varphi$.*

Proof. We prove this by induction on formula complexity. The base case is given by Lemma 6.38, so we move to the induction step. For simplicity we split the case of negation into subcases depending on the structure of the formula to which the negation is applied.

1. Suppose $(\varphi \wedge \psi)$ is true in $\mathfrak{N}$, so φ and ψ are true in $\mathfrak{N}$. By the induction hypothesis, $\mathbf{Q} \vdash \varphi$ and $\mathbf{Q} \vdash \psi$, so $\mathbf{Q} \vdash (\varphi \wedge \psi)$ by logic.

2. Suppose $\neg(\varphi \wedge \psi)$ is true in $\mathfrak{N}$, so either $\neg\varphi$ or $\neg\psi$ is true in $\mathfrak{N}$. Without loss of generality, suppose the former. By the induction hypothesis $\mathbf{Q} \vdash \neg\varphi$, and hence $\mathbf{Q} \vdash \neg(\varphi \wedge \psi)$ by logic.

3. Suppose $(\varphi \vee \psi)$ is true in $\mathfrak{N}$, so either φ is true in $\mathfrak{N}$ or ψ is true in $\mathfrak{N}$. Without loss of generality, suppose the former holds. By the induction hypothesis $\mathbf{Q} \vdash \varphi$, and hence $\mathbf{Q} \vdash (\varphi \vee \psi)$ by logic.

4. Suppose $\neg(\varphi \vee \psi)$ is true in $\mathfrak{N}$, so $\neg\varphi$ and $\neg\psi$ are true in $\mathfrak{N}$. Then $\mathbf{Q} \vdash \neg\varphi$ and $\mathbf{Q} \vdash \neg\psi$ by the induction hypothesis. Consequently, $\mathbf{Q} \vdash \neg(\varphi \vee \psi)$ by logic.

5. Suppose that $(\forall x < t)\ \varphi(x)$ is true in $\mathfrak{N}$, where t is a closed term and $k = \mathrm{Val}^{\mathfrak{N}}(t)$. By the induction hypothesis and logic, if $\varphi(\overline{n})$ is true in $\mathfrak{N}$ for all $n < \mathrm{Val}^{\mathfrak{N}}(t)$ then $\mathbf{Q} \vdash \varphi(\overline{0}) \wedge \cdots \wedge \varphi(\overline{k-1})$. By Lemma 6.39 it follows that $\mathbf{Q} \vdash (\forall x < t)\ \varphi(x)$.

6. The case for the bounded existential quantifier, where we have a sentence of the form $(\exists x < t)\ \varphi(x)$, is similar to that for the bounded universal quantifier.

7. Suppose that $\neg(\forall x < t)\ \varphi(x)$ is true in $\mathfrak{N}$, where t is a closed term. This sentence is equivalent to the sentence $(\exists x < t)\ \neg\varphi(x)$, with the equivalence derivable in $\mathbf{Q}$, so we may apply the reasoning for bounded existential quantifiers.

8. Similarly, suppose that $\neg(\exists x < t)\ \varphi(x)$ is true in $\mathfrak{N}$, where t is a closed term. This sentence is equivalent in $\mathbf{Q}$ to $(\forall x < t)\ \neg\varphi(x)$, and so we may apply the reasoning for bounded universal quantifiers.

9. Finally, suppose $\neg\varphi$ is true in $\mathfrak{N}$. The only cases remaining are when φ is atomic and when $\neg\varphi \equiv \neg\neg\psi$ for some Δ_0 sentence ψ. If φ is atomic then by Lemma 6.38, $\mathbf{Q} \vdash \neg\varphi$. If $\neg\varphi \equiv \neg\neg\psi$, then by logic it is provably equivalent in $\mathbf{Q}$ to ψ, which is true in $\mathfrak{N}$ since $\neg\varphi$ is true in $\mathfrak{N}$. By the induction hypothesis we therefore have that $\mathbf{Q} \vdash \neg\varphi$.

□

Theorem 6.41 (Σ_1-Completeness). *If φ is a Σ_1 sentence which is true in $\mathfrak{N}$, then $\mathbf{Q} \vdash \varphi$.*

Proof. If $\exists x \varphi(x)$ is a Σ_1 sentence which is true in $\mathfrak{N}$, then there exists a natural number n and a variable assignment s such that $s(x) = n$ and $\mathfrak{N}, s \vDash \varphi(x)$. By standard facts about the satisfaction relation it follows that $\mathfrak{N} \vDash \varphi(\overline{n})$. But $\varphi(\overline{n})$ is a Δ_0 formula, so by Lemma 6.40 we have that $\mathbf{Q} \vdash \varphi(\overline{n})$, and hence by logic we also have that $\mathbf{Q} \vdash \exists x\, \varphi(x)$. □

6.5.3 Gödel's First Incompleteness Theorem

We can now describe Gödel's original proof of the first incompleteness theorem. Let **T** be any computably axiomatized theory in a language extending the language of arithmetic, such that **T** includes the axioms of **Q**. This means that, in particular, **T** represents computable functions and relations.

We have argued that, given a reasonable coding of formulas and proofs as numbers, the relation $\mathrm{Prf}_{\mathbf{T}}(x, y)$ is computable, where $\mathrm{Prf}_{\mathbf{T}}(x, y)$ holds if and only if x is the Gödel number of a derivation of the formula with Gödel number y in **T**. In fact, for the particular theory that Gödel had in mind, Gödel was able to show that this relation is primitive recursive, using the list of 45 functions and relations in his paper. The 45th relation, xBy, is just $\mathrm{Prf}_{\mathbf{T}}(x, y)$ for his particular choice of **T**. Remember that where Gödel uses the word "recursive" in his paper, we would now use the phrase "primitive recursive."

Since $\mathrm{Prf}_{\mathbf{T}}(x, y)$ is computable, it is representable in **T**. We will use $\mathsf{Prf}_{\mathbf{T}}(x, y)$ to refer to the formula that represents it. Let $\mathsf{Prov}_{\mathbf{T}}(y)$ be the formula $\exists x\, \mathsf{Prf}_{\mathbf{T}}(x, y)$. This describes the 46th relation, $\mathrm{Bew}(y)$, on Gödel's list. As Gödel notes, this is the only relation that "cannot be asserted to be recursive." What he probably meant is this: from the definition, it is not clear that it is computable; and later developments, in fact, show that it is not.

Let **T** be an axiomatizable theory containing **Q**. Then $\mathrm{Prf}_{\mathbf{T}}(x, y)$ is decidable, hence representable in **Q** by a formula $\mathsf{Prf}_{\mathbf{T}}(x, y)$. Let $\mathsf{Prov}_{\mathbf{T}}(y)$ be the formula we described above. By the fixed-point lemma, there is a formula $\gamma_{\mathbf{T}}$ such that **Q** (and hence **T**) derives

$$\gamma_{\mathbf{T}} \leftrightarrow \neg \mathsf{Prov}_{\mathbf{T}}(\ulcorner \gamma_{\mathbf{T}} \urcorner). \tag{6.3}$$

Note that $\gamma_{\mathbf{T}}$ says, in essence, "$\gamma_{\mathbf{T}}$ is not derivable in **T**."

Lemma 6.42. *If **T** is a consistent, axiomatizable theory extending **Q**, then $\mathbf{T} \nvdash \gamma_{\mathbf{T}}$.*

Proof. Suppose **T** derives $\gamma_{\mathbf{T}}$. Then there *is* a derivation, and so, for some number m, the relation $\mathrm{Prf}_{\mathbf{T}}(m, {}^{\#}\gamma_{\mathbf{T}}{}^{\#})$ holds. But then **Q** derives the sentence $\mathsf{Prf}_{\mathbf{T}}(\overline{m}, \ulcorner \gamma_{\mathbf{T}} \urcorner)$. So **Q** derives $\exists x\, \mathsf{Prf}_{\mathbf{T}}(x, \ulcorner \gamma_{\mathbf{T}} \urcorner)$, which is, by definition, $\mathsf{Prov}_{\mathbf{T}}(\ulcorner \gamma_{\mathbf{T}} \urcorner)$.

By (6.3), **Q** derives $\neg\gamma_{\mathbf{T}}$, and since **T** extends **Q**, so does **T**. We have shown that if **T** derives $\gamma_{\mathbf{T}}$, then it also derives $\neg\gamma_{\mathbf{T}}$, and hence it would be inconsistent. □

Definition 6.43 (ω-consistency)**.** A theory **T** is *ω-consistent* if the following holds: if $\exists x\, \varphi(x)$ is any sentence and **T** derives $\neg\varphi(\overline{0})$, $\neg\varphi(\overline{1})$, $\neg\varphi(\overline{2})$, ... then **T** does not prove $\exists x\, \varphi(x)$.

Note that every ω-consistent theory is also consistent. This follows simply from the fact that if **T** is inconsistent, then $\mathbf{T} \vdash \varphi$ for every φ. In particular, if **T** is inconsistent, it derives both $\neg\varphi(\overline{n})$ for every n and also derives $\exists x\, \varphi(x)$. So, if **T** is inconsistent, it is ω-inconsistent. By contraposition, if **T** is ω-consistent, it must be consistent.

Lemma 6.44. *If* **T** *is an ω-consistent, axiomatizable theory extending* **Q**, *then* $\mathbf{T} \nvdash \neg\gamma_{\mathbf{T}}$.

Proof. We show that if **T** derives $\neg\gamma_{\mathbf{T}}$, then it is ω-inconsistent. Suppose **T** derives $\neg\gamma_{\mathbf{T}}$. If **T** is inconsistent, it is ω-inconsistent, and we are done. Otherwise, **T** is consistent, so it does not derive $\gamma_{\mathbf{T}}$ by Lemma 6.42. Since there is no derivation of $\gamma_{\mathbf{T}}$ in **T**, **Q** derives

$$\neg\mathsf{Prf}_{\mathbf{T}}(\overline{0}, \ulcorner\gamma_{\mathbf{T}}\urcorner), \neg\mathsf{Prf}_{\mathbf{T}}(\overline{1}, \ulcorner\gamma_{\mathbf{T}}\urcorner), \neg\mathsf{Prf}_{\mathbf{T}}(\overline{2}, \ulcorner\gamma_{\mathbf{T}}\urcorner), \ldots$$

and so does **T**. On the other hand, by (6.3), $\neg\gamma_{\mathbf{T}}$ is equivalent to $\exists x\, \mathsf{Prf}_{\mathbf{T}}(x, \ulcorner\gamma_{\mathbf{T}}\urcorner)$. So **T** is ω-inconsistent. □

Theorem 6.45 (First Incompleteness Theorem — Gödel's version)**.** *Let* **T** *be any ω-consistent, axiomatizable theory extending* **Q**. *Then* **T** *is not complete.*

Proof. If **T** is ω-consistent, it is consistent, so $\mathbf{T} \nvdash \gamma_{\mathbf{T}}$ by Lemma 6.42. By Lemma 6.44, $\mathbf{T} \nvdash \neg\gamma_{\mathbf{T}}$. This means that **T** is incomplete, since it derives neither $\gamma_{\mathbf{T}}$ nor $\neg\gamma_{\mathbf{T}}$. □

Remark 32 (Computability-theoretic proof)*.* There is an alternative, more direct proof of the First Incompleteness Theorem via computability theory: if **T** were a complete, consistent, axiomatizable extension of **Q**, then **T** would be decidable, contradicting the undecidability of **Q** (see Theorem 6.55 below). This computability-theoretic argument avoids the need for the ω-consistency hypothesis entirely, though it does not construct an explicit independent sentence.

6.5.4 Rosser's Theorem

Can we modify Gödel's proof to get a stronger result, replacing "ω-consistent" with simply "consistent"? The answer is "yes," using a trick discovered by Rosser. Rosser's trick is to use a "modified" derivability predicate $\mathsf{RProv}_T(y)$ instead of $\mathsf{Prov}_{\mathbf{T}}(y)$.

Theorem 6.46 (Rosser's Theorem). *Let* $\mathbf{T}$ *be any consistent, axiomatizable theory extending* $\mathbf{Q}$*. Then* $\mathbf{T}$ *is not complete.*

Proof idea. Define a "Rosser provability predicate" $\mathsf{RProv}_{\mathbf{T}}(y)$ asserting "y has a proof shorter than any refutation of y." Apply the fixed-point lemma to obtain a sentence ρ asserting its own non-Rosser-provability. Consistency alone (without ω-consistency) suffices to show $\mathbf{T} \nvdash \rho$ and $\mathbf{T} \nvdash \neg\rho$.

Proof. Recall that $\mathsf{Prov}_{\mathbf{T}}(y)$ is defined as $\exists x\, \mathsf{Prf}_{\mathbf{T}}(x, y)$, where $\mathsf{Prf}_{\mathbf{T}}(x, y)$ represents the decidable relation which holds iff x is the Gödel number of a derivation of the sentence with Gödel number y. The relation that holds between x and y if x is the Gödel number of a *refutation* of the sentence with Gödel number y is also decidable. Let $\mathrm{not}(x)$ be the primitive recursive function which does the following: if x is the code of a formula φ, $\mathrm{not}(x)$ is a code of $\neg\varphi$. Then $\mathrm{Ref}_{\mathbf{T}}(x, y)$ holds iff $\mathrm{Prf}_{\mathbf{T}}(x, \mathrm{not}(y))$. Let $\mathsf{Ref}_{\mathbf{T}}(x, y)$ represent it. Then, if $\mathbf{T} \vdash \neg\varphi$ and δ is a corresponding derivation, $\mathbf{Q} \vdash \mathsf{Ref}_{\mathbf{T}}(\ulcorner\delta\urcorner, \ulcorner\varphi\urcorner)$. We define $\mathsf{RProv}_{\mathbf{T}}(y)$ as

$$\exists x\, (\mathsf{Prf}_{\mathbf{T}}(x, y) \land \forall z\, (z < x \to \neg\mathsf{Ref}_{\mathbf{T}}(z, y))).$$

Roughly, $\mathsf{RProv}_{\mathbf{T}}(y)$ says "there is a proof of y in $\mathbf{T}$, and there is no shorter refutation of y." Assuming $\mathbf{T}$ is consistent, $\mathsf{RProv}_{\mathbf{T}}(y)$ is true of the same numbers as $\mathsf{Prov}_{\mathbf{T}}(y)$; but from the point of view of *provability* in $\mathbf{T}$ (and we now know that there is a difference between truth and provability!) the two have different properties. If $\mathbf{T}$ is *in*consistent, then the two do *not* hold of the same numbers!

By the fixed-point lemma, there is a formula $\rho_{\mathbf{T}}$ such that

$$\mathbf{Q} \vdash \rho_{\mathbf{T}} \leftrightarrow \neg\mathsf{RProv}_{\mathbf{T}}(\ulcorner\rho_{\mathbf{T}}\urcorner). \tag{6.4}$$

In contrast to the proof of Theorem 6.45, here we claim that if $\mathbf{T}$ is consistent, $\mathbf{T}$ does not derive $\rho_{\mathbf{T}}$, and $\mathbf{T}$ also does not derive $\neg\rho_{\mathbf{T}}$. (In other words, we do not need the assumption of ω-consistency.)

First, let us show that $\mathbf{T} \nvdash \rho_{\mathbf{T}}$. Suppose it did, so there is a derivation of $\rho_{\mathbf{T}}$ from T; let n be its Gödel number. Then $\mathbf{Q} \vdash \mathsf{Prf}_{\mathbf{T}}(\overline{n}, \ulcorner\rho_{\mathbf{T}}\urcorner)$, since $\mathsf{Prf}_{\mathbf{T}}$ represents $\mathrm{Prf}_{\mathbf{T}}$ in $\mathbf{Q}$. Also, for each $k < n$, k is not the Gödel number of a derivation of $\neg\rho_{\mathbf{T}}$, since $\mathbf{T}$ is consistent. So for each $k < n$, $\mathbf{Q} \vdash \neg\mathsf{Ref}_{\mathbf{T}}(\overline{k}, \ulcorner\rho_{\mathbf{T}}\urcorner)$. By properties of $<$ in $\mathbf{Q}$, $\mathbf{Q} \vdash \forall z\, (z < \overline{n} \to \neg\mathsf{Ref}_{\mathbf{T}}(z, \ulcorner\rho_{\mathbf{T}}\urcorner))$. Thus,

$$\mathbf{Q} \vdash \exists x\, (\mathsf{Prf}_{\mathbf{T}}(x, \ulcorner\rho_{\mathbf{T}}\urcorner) \land \forall z\, (z < x \to \neg\mathsf{Ref}_{\mathbf{T}}(z, \ulcorner\rho_{\mathbf{T}}\urcorner))),$$

but that is just $\mathsf{RProv}_{\mathbf{T}}(\ulcorner\rho_{\mathbf{T}}\urcorner)$. By (6.4), $\mathbf{Q} \vdash \lnot\rho_{\mathbf{T}}$. Since $\mathbf{T}$ extends $\mathbf{Q}$, also $\mathbf{T} \vdash \lnot\rho_{\mathbf{T}}$. We have assumed that $\mathbf{T} \vdash \rho_{\mathbf{T}}$, so $\mathbf{T}$ would be inconsistent, contrary to the assumption of the theorem.

Now, let us show that $\mathbf{T} \nvdash \lnot\rho_{\mathbf{T}}$. Again, suppose it did, and suppose n is the Gödel number of a derivation of $\lnot\rho_{\mathbf{T}}$. Then $\mathrm{Ref}_{\mathbf{T}}(n, {}^{\#}\rho_{\mathbf{T}}{}^{\#})$ holds, and since $\mathsf{Ref}_{\mathbf{T}}$ represents $\mathrm{Ref}_{\mathbf{T}}$ in $\mathbf{Q}$, $\mathbf{Q} \vdash \mathsf{Ref}_{\mathbf{T}}(\overline{n}, \ulcorner\rho_{\mathbf{T}}\urcorner)$. We will again show that $\mathbf{T}$ would then be inconsistent because it would also derive $\rho_{\mathbf{T}}$. Since

$$\mathbf{Q} \vdash \rho_{\mathbf{T}} \leftrightarrow \lnot\mathsf{RProv}_{\mathbf{T}}(\ulcorner\rho_{\mathbf{T}}\urcorner),$$

and since $\mathbf{T}$ extends $\mathbf{Q}$, it suffices to show that

$$\mathbf{Q} \vdash \lnot\mathsf{RProv}_{\mathbf{T}}(\ulcorner\rho_{\mathbf{T}}\urcorner).$$

The sentence $\lnot\mathsf{RProv}_{\mathbf{T}}(\ulcorner\rho_{\mathbf{T}}\urcorner)$, i.e.,

$$\lnot\exists x\,(\mathsf{Prf}_{\mathbf{T}}(x, \ulcorner\rho_{\mathbf{T}}\urcorner) \land \forall z\,(z < x \to \lnot\mathsf{Ref}_{\mathbf{T}}(z, \ulcorner\rho_{\mathbf{T}}\urcorner))),$$

is logically equivalent to

$$\forall x\,(\mathsf{Prf}_{\mathbf{T}}(x, \ulcorner\rho_{\mathbf{T}}\urcorner) \to \exists z\,(z < x \land \mathsf{Ref}_{\mathbf{T}}(z, \ulcorner\rho_{\mathbf{T}}\urcorner))).$$

We argue informally using logic, making use of facts about what $\mathbf{Q}$ derives. Suppose x is arbitrary and $\mathsf{Prf}_{\mathbf{T}}(x, \ulcorner\rho_{\mathbf{T}}\urcorner)$. We already know that $\mathbf{T} \nvdash \rho_{\mathbf{T}}$, and so for every k, $\mathbf{Q} \vdash \lnot\mathsf{Prf}_{\mathbf{T}}(\overline{k}, \ulcorner\rho_{\mathbf{T}}\urcorner)$. Thus, for every k it follows that $x \neq \overline{k}$. In particular, we have (a) that $x \neq \overline{n}$. We also have $\lnot(x = \overline{0} \lor x = \overline{1} \lor \dots \lor x = \overline{n-1})$ and so by properties of $<$ in $\mathbf{Q}$, (b) $\lnot(x < \overline{n})$. By trichotomy in $\mathbf{Q}$, $\overline{n} < x$. Since $\mathbf{Q} \vdash \mathsf{Ref}_{\mathbf{T}}(\overline{n}, \ulcorner\rho_{\mathbf{T}}\urcorner)$, we have $\overline{n} < x \land \mathsf{Ref}_{\mathbf{T}}(\overline{n}, \ulcorner\rho_{\mathbf{T}}\urcorner)$, and from that $\exists z\,(z < x \land \mathsf{Ref}_{\mathbf{T}}(z, \ulcorner\rho_{\mathbf{T}}\urcorner))$. Since x was arbitrary we get, as required, that

$$\forall x\,(\mathsf{Prf}_{\mathbf{T}}(x, \ulcorner\rho_{\mathbf{T}}\urcorner) \to \exists z\,(z < x \land \mathsf{Ref}_{\mathbf{T}}(z, \ulcorner\rho_{\mathbf{T}}\urcorner))).$$

□

6.6 The Second Incompleteness Theorem

Gödel's Second Incompleteness Theorem shows that no consistent, sufficiently strong theory can prove its own consistency. The proof rests on the derivability conditions for the provability predicate. We also present Löb's Theorem, which gives a sharp characterization of when a reflection principle can be derived.

6.6.1 The Derivability Conditions for PA

Peano arithmetic, or **PA**, is the theory extending **Q** with induction axioms for all formulas. In other words, one adds to **Q** axioms of the form

$$(\varphi(0) \land \forall x\,(\varphi(x) \to \varphi(x'))) \to \forall x\,\varphi(x)$$

for every formula φ. Notice that this is really a *schema*, which is to say, infinitely many axioms (and it turns out that **PA** is *not* finitely axiomatizable). But since one can effectively determine whether or not a string of symbols is an instance of an induction axiom, the set of axioms for **PA** is computable. **PA** is a much more robust theory than **Q**. For example, one can easily prove that addition and multiplication are commutative, using induction in the usual way. In fact, most finitary number-theoretic and combinatorial arguments can be carried out in **PA**.

Since **PA** is computably axiomatized, the derivability predicate $\mathsf{Prf}_{\mathbf{PA}}(x, y)$ is computable and hence represented in **Q** (and so, in **PA**). As before, we will take $\mathsf{Prf}_{\mathbf{PA}}(x, y)$ to denote the formula representing the relation. Let $\mathsf{Prov}_{\mathbf{PA}}(y)$ be the formula $\exists x\, \mathsf{Prf}_{\mathbf{PA}}(x, y)$, which intuitively says, "y is derivable from the axioms of **PA**." The reason we need a little bit more than the axioms of **Q** is we need to know that the theory we are using is strong enough to derive a few basic facts about this derivability predicate. In fact, what we need are the following facts:

Definition 6.47 (Derivability Conditions)**.** A formula $\mathsf{Prov}_{\mathbf{T}}(y)$ satisfies the *Hilbert–Bernays–Löb derivability conditions* for a theory **T** if the following hold:

P1. If $\mathbf{T} \vdash \varphi$, then $\mathbf{T} \vdash \mathsf{Prov}_{\mathbf{T}}(\ulcorner \varphi \urcorner)$.

P2. For all formulas φ and ψ,

$$\mathbf{T} \vdash \mathsf{Prov}_{\mathbf{T}}(\ulcorner \varphi \to \psi \urcorner) \to (\mathsf{Prov}_{\mathbf{T}}(\ulcorner \varphi \urcorner) \to \mathsf{Prov}_{\mathbf{T}}(\ulcorner \psi \urcorner)).$$

P3. For every formula φ,

$$\mathbf{T} \vdash \mathsf{Prov}_{\mathbf{T}}(\ulcorner \varphi \urcorner) \to \mathsf{Prov}_{\mathbf{T}}(\ulcorner \mathsf{Prov}_{\mathbf{T}}(\ulcorner \varphi \urcorner) \urcorner).$$

The only way to verify that these three properties hold is to describe the formula $\mathsf{Prov}_{\mathbf{PA}}(y)$ carefully and use the axioms of **PA** to describe the relevant formal derivations. Conditions (1) and (2) are easy; it is really condition (3) that requires work. Carrying out the details would be tedious and uninteresting, so here we will ask you to take it on faith that **PA** has the three properties listed above. A reasonable choice of $\mathsf{Prov}_{\mathbf{PA}}(y)$ will also satisfy

P4. If $\mathbf{PA} \vdash \mathsf{Prov}_{\mathbf{PA}}(\ulcorner \varphi \urcorner)$, then $\mathbf{PA} \vdash \varphi$.

But we will not need this fact.

6.6.2 The Second Incompleteness Theorem

How can we express the assertion that **PA** does not prove its own consistency? Saying **PA** is inconsistent amounts to saying that $\mathbf{PA} \vdash 0 = 1$. So we can take the consistency statement $\mathsf{Con}_{\mathbf{PA}}$ to be the sentence $\neg\mathsf{Prov}_{\mathbf{PA}}(\ulcorner 0 = 1 \urcorner)$, and then the following theorem does the job:

Theorem 6.48 (Second Incompleteness Theorem)**.** *Assuming* **PA** *is consistent, then* **PA** *does not derive* $\mathsf{Con}_{\mathbf{PA}}$.

It is important to note that the theorem depends on the particular representation of $\mathsf{Con}_{\mathbf{PA}}$ (i.e., the particular representation of $\mathsf{Prov}_{\mathbf{PA}}(y)$). All we will use is that the representation of $\mathsf{Prov}_{\mathbf{PA}}(y)$ satisfies the three derivability conditions, so the theorem generalizes to any theory with a derivability predicate having these properties.

It is informative to read Gödel's sketch of an argument, since the theorem follows like a good punch line. It goes like this. Let $\gamma_{\mathbf{PA}}$ be the Gödel sentence that we constructed in the proof of Theorem 6.45. We have shown "If **PA** is consistent, then **PA** does not derive $\gamma_{\mathbf{PA}}$." If we formalize this *in* **PA**, we have a proof of

$$\mathsf{Con}_{\mathbf{PA}} \to \neg\mathsf{Prov}_{\mathbf{PA}}(\ulcorner \gamma_{\mathbf{PA}} \urcorner).$$

Now suppose **PA** derives $\mathsf{Con}_{\mathbf{PA}}$. Then it derives $\neg\mathsf{Prov}_{\mathbf{PA}}(\ulcorner \gamma_{\mathbf{PA}} \urcorner)$. But since $\gamma_{\mathbf{PA}}$ is a Gödel sentence, this is equivalent to $\gamma_{\mathbf{PA}}$. So **PA** derives $\gamma_{\mathbf{PA}}$.

But: we know that if **PA** is consistent, it does not derive $\gamma_{\mathbf{PA}}$! So if **PA** is consistent, it cannot derive $\mathsf{Con}_{\mathbf{PA}}$.

To make the argument more precise, we will let $\gamma_{\mathbf{PA}}$ be the Gödel sentence for **PA** and use the derivability conditions (P1)–(P3) to show that **PA** derives $\mathsf{Con}_{\mathbf{PA}} \to \gamma_{\mathbf{PA}}$. This will show that **PA** does not derive $\mathsf{Con}_{\mathbf{PA}}$. Here is a sketch of the proof, in **PA**. (For simplicity, we drop the **PA** subscripts.)

$$\gamma \leftrightarrow \neg\mathsf{Prov}(\ulcorner\gamma\urcorner) \tag{6.5}$$
γ is a Gödel sentence

$$\gamma \to \neg\mathsf{Prov}(\ulcorner\gamma\urcorner) \tag{6.6}$$
from (6.5)

$$\gamma \to (\mathsf{Prov}(\ulcorner\gamma\urcorner) \to \bot) \tag{6.7}$$
from (6.6) by logic

$$\mathsf{Prov}(\ulcorner\gamma \to (\mathsf{Prov}(\ulcorner\gamma\urcorner) \to \bot)\urcorner) \tag{6.8}$$
from (6.7) by condition P1

$$\mathsf{Prov}(\ulcorner\gamma\urcorner) \to \mathsf{Prov}(\ulcorner(\mathsf{Prov}(\ulcorner\gamma\urcorner) \to \bot)\urcorner) \tag{6.9}$$
from (6.8) by condition P2

$$\mathsf{Prov}(\ulcorner\gamma\urcorner) \to (\mathsf{Prov}(\ulcorner\mathsf{Prov}(\ulcorner\gamma\urcorner)\urcorner) \to \mathsf{Prov}(\ulcorner\bot\urcorner)) \tag{6.10}$$
from (6.9) by condition P2 and logic

$$\mathsf{Prov}(\ulcorner\gamma\urcorner) \to \mathsf{Prov}(\ulcorner\mathsf{Prov}(\ulcorner\gamma\urcorner)\urcorner) \tag{6.11}$$
by P3

$$\mathsf{Prov}(\ulcorner\gamma\urcorner) \to \mathsf{Prov}(\ulcorner\bot\urcorner) \tag{6.12}$$
from (6.10) and (6.11) by logic

$$\mathsf{Con} \to \neg\mathsf{Prov}(\ulcorner\gamma\urcorner) \tag{6.13}$$
contraposition of (6.12) and $\mathsf{Con} \equiv \neg\mathsf{Prov}(\ulcorner\bot\urcorner)$

$$\mathsf{Con} \to \gamma$$
from (6.5) and (6.13) by logic

The use of logic in the above involves just elementary facts from propositional logic, e.g., (6.7) uses $\vdash \neg\varphi \leftrightarrow (\varphi \to \bot)$ and (6.12) uses $\varphi \to (\psi \to \chi), \varphi \to \psi \vdash \varphi \to \chi$. The use of condition P2 in (6.9) and (6.10) relies on instances of P2, $\mathsf{Prov}(\ulcorner\varphi \to \psi\urcorner) \to (\mathsf{Prov}(\ulcorner\varphi\urcorner) \to \mathsf{Prov}(\ulcorner\psi\urcorner))$. In the first one, $\varphi \equiv \gamma$ and $\psi \equiv \mathsf{Prov}(\ulcorner\gamma\urcorner) \to \bot$; in the second, $\varphi \equiv \mathsf{Prov}(\ulcorner G\urcorner)$ and $\psi \equiv \bot$.

The more abstract version of the second incompleteness theorem is as follows:

Theorem 6.49 (Second Incompleteness Theorem — general version)**.** *Let* **T** *be any consistent, axiomatized theory extending* **Q** *and let* $\mathsf{Prov}_{\mathbf{T}}(y)$ *be any formula satisfying derivability conditions P1–P3 for* **T**. *Then* **T** *does not derive* Con_T.

The moral of the story is that no "reasonable" consistent theory for mathematics can derive its own consistency statement. Suppose **T** is a theory of mathematics that includes **Q** and Hilbert's "finitary" reasoning (whatever that may be). Then, the whole of **T** cannot derive the consistency statement of **T**, and so, a fortiori, the finitary fragment cannot derive the consistency statement of **T** either. In that sense, there cannot be a finitary consistency proof for "all of mathematics."

There is some leeway in interpreting the term "finitary," and Gödel, in the 1931 paper, grants the possibility that something we may consider "finitary" may lie outside the kinds of mathematics Hilbert wanted to formalize. But Gödel was being charitable; today, it is hard to see how we might find something that can reasonably be called finitary but is not formalizable in, say, **ZFC**, Zermelo–Fraenkel set theory with the axiom of choice.

6.6.3 Löb's Theorem

The Gödel sentence for a theory **T** is a fixed point of $\neg\mathsf{Prov}_{\mathbf{T}}(y)$, i.e., a sentence γ such that

$$\mathbf{T} \vdash \neg\mathsf{Prov}_{\mathbf{T}}(\ulcorner\gamma\urcorner) \leftrightarrow \gamma.$$

It is not derivable, because if $\mathbf{T} \vdash \gamma$, (a) by derivability condition (1), $\mathbf{T} \vdash \mathsf{Prov}_{\mathbf{T}}(\ulcorner\gamma\urcorner)$, and (b) $\mathbf{T} \vdash \gamma$ together with $\mathbf{T} \vdash \neg\mathsf{Prov}_{\mathbf{T}}(\ulcorner\gamma\urcorner) \leftrightarrow \gamma$ gives $\mathbf{T} \vdash \neg\mathsf{Prov}_{\mathbf{T}}(\ulcorner\gamma\urcorner)$, and so **T** would be inconsistent. Now it is natural to ask about the status of a fixed point of $\mathsf{Prov}_{\mathbf{T}}(y)$, i.e., a sentence δ such that

$$\mathbf{T} \vdash \mathsf{Prov}_{\mathbf{T}}(\ulcorner\delta\urcorner) \leftrightarrow \delta.$$

If it were derivable, $\mathbf{T} \vdash \mathsf{Prov}_{\mathbf{T}}(\ulcorner\delta\urcorner)$ by condition (1), but the same conclusion follows if we apply modus ponens to the equivalence above. Hence, we do not get that **T** is inconsistent, at least not by the same argument as in the case of the Gödel sentence. This of course does not show that **T** *does* derive δ.

We can make headway on this question if we generalize it a bit. The left-to-right direction of the fixed point equivalence, $\mathsf{Prov}_{\mathbf{T}}(\ulcorner\delta\urcorner) \rightarrow \delta$, is an instance of a general schema called a *reflection principle*: $\mathsf{Prov}_{\mathbf{T}}(\ulcorner\varphi\urcorner) \rightarrow \varphi$. It is called that because it expresses, in a sense, that **T** can "reflect" about what it can derive; basically it says, "If **T** can derive φ, then φ is true," for any φ. This is true for sound theories only, of course, and this suggests that theories will in general not derive every instance of it. So which instances can a theory (strong enough, and satisfying the derivability conditions) derive? Certainly all those where φ itself is derivable. And that is it, as the next result shows.

The heuristic for the proof of Löb's theorem is a clever proof that Santa Claus exists. (If you do not like that conclusion, you are free to substitute any other conclusion you would like.) Here it is:

1. Let X be the sentence, "If X is true, then Santa Claus exists."
2. Suppose X is true.
3. Then what it says holds; i.e., we have: if X is true, then Santa Claus exists.

4. Since we are assuming X is true, we can conclude that Santa Claus exists, by modus ponens from (2) and (3).

5. We have succeeded in deriving (4), "Santa Claus exists," from the assumption (2), "X is true." By conditional proof, we have shown: "If X is true, then Santa Claus exists."

6. But this is just the sentence X. So we have shown that X is true.

7. But then, by the argument (2)–(4) above, Santa Claus exists.

A formalization of this idea, replacing "is true" with "is derivable," and "Santa Claus exists" with φ, yields the proof of Löb's theorem. The trick is to apply the fixed-point lemma to the formula $\mathsf{Prov}_{\mathbf{T}}(y) \rightarrow \varphi$. The fixed point of that corresponds to the sentence X in the preceding sketch.

Theorem 6.50 (Löb's Theorem)**.** *Let* $\mathbf{T}$ *be an axiomatizable theory extending* $\mathbf{Q}$*, and suppose* $\mathsf{Prov}_{\mathbf{T}}(y)$ *is a formula satisfying conditions P1–P3 (Definition 6.47). If* $\mathbf{T}$ *derives* $\mathsf{Prov}_{\mathbf{T}}(\ulcorner\varphi\urcorner) \rightarrow \varphi$*, then in fact* $\mathbf{T}$ *derives* φ*.*

Put differently, if $\mathbf{T} \nvdash \varphi$, then $\mathbf{T} \nvdash \mathsf{Prov}_{\mathbf{T}}(\ulcorner\varphi\urcorner) \rightarrow \varphi$.

Proof. Suppose φ is a sentence such that $\mathbf{T}$ derives $\mathsf{Prov}_{\mathbf{T}}(\ulcorner\varphi\urcorner) \rightarrow \varphi$. Let $\psi(y)$ be the formula $\mathsf{Prov}_{\mathbf{T}}(y) \rightarrow \varphi$, and use the fixed-point lemma to find a sentence θ such that $\mathbf{T}$ derives $\theta \leftrightarrow \psi(\ulcorner\theta\urcorner)$. Then each of the following is derivable in **T**:

$$\theta \leftrightarrow (\mathsf{Prov}_{\mathbf{T}}(\ulcorner\theta\urcorner) \to \varphi) \tag{6.14}$$

θ is a fixed point of $\psi(y)$

$$\theta \to (\mathsf{Prov}_{\mathbf{T}}(\ulcorner\theta\urcorner) \to \varphi) \tag{6.15}$$

from (6.14)

$$\mathsf{Prov}_{\mathbf{T}}(\ulcorner\theta \to (\mathsf{Prov}_{\mathbf{T}}(\ulcorner\theta\urcorner) \to \varphi)\urcorner) \tag{6.16}$$

from (6.15) by condition P1

$$\mathsf{Prov}_{\mathbf{T}}(\ulcorner\theta\urcorner) \to \mathsf{Prov}_{\mathbf{T}}(\ulcorner\mathsf{Prov}_{\mathbf{T}}(\ulcorner\theta\urcorner) \to \varphi\urcorner) \tag{6.17}$$

from (6.16) using condition P2

$$\mathsf{Prov}_{\mathbf{T}}(\ulcorner\theta\urcorner) \to (\mathsf{Prov}_{\mathbf{T}}(\ulcorner\mathsf{Prov}_{\mathbf{T}}(\ulcorner\theta\urcorner)\urcorner) \to \mathsf{Prov}_{\mathbf{T}}(\ulcorner\varphi\urcorner)) \tag{6.18}$$

from (6.17) using P2 again

$$\mathsf{Prov}_{\mathbf{T}}(\ulcorner\theta\urcorner) \to \mathsf{Prov}_{\mathbf{T}}(\ulcorner\mathsf{Prov}_{\mathbf{T}}(\ulcorner\theta\urcorner)\urcorner) \tag{6.19}$$

by derivability condition P3

$$\mathsf{Prov}_{\mathbf{T}}(\ulcorner\theta\urcorner) \to \mathsf{Prov}_{\mathbf{T}}(\ulcorner\varphi\urcorner) \tag{6.20}$$

from (6.18) and (6.19)

$$\mathsf{Prov}_{\mathbf{T}}(\ulcorner\varphi\urcorner) \to \varphi \tag{6.21}$$

by assumption of the theorem

$$\mathsf{Prov}_{\mathbf{T}}(\ulcorner\theta\urcorner) \to \varphi \tag{6.22}$$

from (6.20) and (6.21)

$$(\mathsf{Prov}_{\mathbf{T}}(\ulcorner\theta\urcorner) \to \varphi) \to \theta \tag{6.23}$$

from (6.14)

$$\theta \tag{6.24}$$

from (6.22) and (6.23)

$$\mathsf{Prov}_{\mathbf{T}}(\ulcorner\theta\urcorner) \tag{6.25}$$

from (6.24) by condition P1

φ from (6.21) and (6.25)

□

With Löb's theorem in hand, there is a short proof of the second incompleteness theorem (for theories having a derivability predicate satisfying conditions P1–P3): if $\mathbf{T} \vdash \mathsf{Prov}_{\mathbf{T}}(\ulcorner\bot\urcorner) \to \bot$, then $\mathbf{T} \vdash \bot$. If **T** is consistent, $\mathbf{T} \nvdash \bot$. So, $\mathbf{T} \nvdash \mathsf{Prov}_{\mathbf{T}}(\ulcorner\bot\urcorner) \to \bot$, i.e., $\mathbf{T} \nvdash \mathsf{Con}_{\mathbf{T}}$. We can also apply it to show that δ, the fixed point of $\mathsf{Prov}_{\mathbf{T}}(x)$, is derivable. For since

$$\mathbf{T} \vdash \mathsf{Prov}_{\mathbf{T}}(\ulcorner\delta\urcorner) \leftrightarrow \delta$$

in particular

$$\mathbf{T} \vdash \mathsf{Prov}_{\mathbf{T}}(\ulcorner \delta \urcorner) \to \delta$$

and so by Löb's theorem, $\mathbf{T} \vdash \delta$.

6.7 The Undefinability of Truth

The notion of *definability* depends on having a formal semantics for the language of arithmetic. We have described a set of formulas and sentences in the language of arithmetic. The "intended interpretation" is to read such sentences as making assertions about the natural numbers, and such an assertion can be true or false. Let $\mathfrak{N}$ be the structure with domain $\mathbb{N}$ and the standard interpretation for the symbols in the language of arithmetic. Then $\mathfrak{N} \vDash \varphi$ means "φ is true in the standard interpretation."

Definition 6.51 (Definability in $\mathfrak{N}$)**.** A relation $R(x_1, \dots, x_k)$ of natural numbers is *definable* in $\mathfrak{N}$ if and only if there is a formula $\varphi(x_1, \dots, x_k)$ in the language of arithmetic such that for every $n_1, \dots, n_k$, $R(n_1, \dots, n_k)$ if and only if $\mathfrak{N} \vDash \varphi(\overline{n}_1, \dots, \overline{n}_k)$.

Put differently, a relation is definable in $\mathfrak{N}$ if and only if it is representable in the theory $\mathbf{TA}$, where $\mathbf{TA} = \{\varphi : \mathfrak{N} \vDash \varphi\}$ is the set of true sentences of arithmetic.

Lemma 6.52. *Every computable relation is definable in $\mathfrak{N}$.*

Proof. It is easy to check that the formula representing a relation in $\mathbf{Q}$ defines the same relation in $\mathfrak{N}$. □

Now one can ask, is the converse also true? That is, is every relation definable in $\mathfrak{N}$ computable? The answer is no. For example:

Lemma 6.53. *The halting relation is definable in $\mathfrak{N}$.*

Proof. Recall that the Kleene normal form theorem states that every partial computable function f has an index e such that $f(x) = U(\mu s\, T(e, x, s))$ for all $x \in \mathbb{N}$, where U and T are primitive recursive and therefore total. Thus, $f(x)$ is defined (i.e., the computation halts) iff there is an s such that $T(e, x, s)$ holds.

Now let H be the halting relation, i.e.,

$$H = \{\langle e, x \rangle : \exists s\, T(e, x, s)\}.$$

Let θ_T define T in $\mathfrak{N}$. Then

$$H = \{\langle e, x\rangle : \mathfrak{N} \models \exists s\, \theta_T(\overline{e}, \overline{x}, s)\},$$

so $\exists s\, \theta_T(z, x, s)$ defines H in $\mathfrak{N}$. □

What about **TA** itself? Is it definable in arithmetic? That is: is the set $\{{}^{\#}\varphi^{\#} : \mathfrak{N} \models \varphi\}$ definable in arithmetic? Tarski's theorem answers this in the negative.

Theorem 6.54 (Tarski's Undefinability Theorem)**.** *The set of true sentences of arithmetic is not definable in arithmetic.*

Proof. Suppose $\theta(x)$ defined it, i.e., $\mathfrak{N} \models \varphi$ iff $\mathfrak{N} \models \theta(\ulcorner\varphi\urcorner)$. By the fixed-point lemma (Lemma 6.34), there is a formula φ such that $\mathbf{Q} \vdash \varphi \leftrightarrow \neg\theta(\ulcorner\varphi\urcorner)$, and hence $\mathfrak{N} \models \varphi \leftrightarrow \neg\theta(\ulcorner\varphi\urcorner)$. But then $\mathfrak{N} \models \varphi$ if and only if $\mathfrak{N} \models \neg\theta(\ulcorner\varphi\urcorner)$, which contradicts the fact that $\theta(y)$ is supposed to define the set of true statements of arithmetic. □

Tarski applied this analysis to a more general philosophical notion of truth. Given any language L, Tarski argued that an adequate notion of truth for L would have to satisfy, for each sentence X,

'X' is true if and only if X.

Tarski's oft-quoted example, for English, is the sentence

'Snow is white' is true if and only if snow is white.

However, for any language strong enough to represent the diagonal function, and any linguistic predicate $T(x)$, we can construct a sentence X satisfying "X if and only if not T('X')." Given that we do not want a truth predicate to declare some sentences to be both true and false, Tarski concluded that one cannot specify a truth predicate for all sentences in a language without, somehow, stepping outside the bounds of the language. In other words, the truth predicate for a language cannot be defined in the language itself.

6.8 Undecidability

We call a theory **T** *undecidable* if there is no computational procedure which, after finitely many steps and unfailingly, provides a correct answer to the question "does **T** prove φ?" for any sentence φ in the language of **T**. So **Q** would be decidable iff there were a computational procedure which decides, given a sentence φ in the language of arithmetic, whether $\mathbf{Q} \vdash \varphi$ or not. We can make this more precise by asking: Is the relation $\mathrm{Prov}_{\mathbf{Q}}(y)$, which holds of y iff y is the Gödel number of a sentence provable in **Q**, recursive? The answer is: no.

Theorem 6.55 (Undecidability of **Q**). *$\mathbf{Q}$ is undecidable, i.e., the relation*

$$\mathrm{Prov}_{\mathbf{Q}}(y) \Leftrightarrow \mathrm{Sent}(y) \wedge \exists x\, \mathrm{Prf}_{\mathbf{Q}}(x, y)$$

is not recursive.

Proof. Suppose it were. Then we could solve the halting problem as follows: Given e and n, we know that $\varphi_e(n) \downarrow$ iff there is an s such that $T(e, n, s)$, where T is Kleene's predicate from the Kleene Normal Form Theorem (see DEF-CMP005, §5.3). Since T is primitive recursive it is representable in $\mathbf{Q}$ by a formula ψ_T, that is, $\mathbf{Q} \vdash \psi_T(\overline{e}, \overline{n}, \overline{s})$ iff $T(e, n, s)$. If $\mathbf{Q} \vdash \psi_T(\overline{e}, \overline{n}, \overline{s})$ then also $\mathbf{Q} \vdash \exists y\, \psi_T(\overline{e}, \overline{n}, y)$. If no such s exists, then $\mathbf{Q} \vdash \neg\psi_T(\overline{e}, \overline{n}, \overline{s})$ for every s. But $\mathbf{Q}$ is ω-consistent, i.e., if $\mathbf{Q} \vdash \neg\varphi(\overline{n})$ for every $n \in \mathbb{N}$, then $\mathbf{Q} \nvdash \exists y\, \varphi(y)$. We know this because the axioms of $\mathbf{Q}$ are true in the standard model $\mathfrak{N}$. So, $\mathbf{Q} \nvdash \exists y\, \psi_T(\overline{e}, \overline{n}, y)$. In other words, $\mathbf{Q} \vdash \exists y\, \psi_T(\overline{e}, \overline{n}, y)$ iff there is an s such that $T(e, n, s)$, i.e., iff $\varphi_e(n) \downarrow$. From e and n we can compute $^{\#}\exists y\, \psi_T(\overline{e}, \overline{n}, y)^{\#}$, let $g(e, n)$ be the primitive recursive function which does that. So

$$h(e, n) = \begin{cases} 1 & \text{if } \mathrm{Prov}_{\mathbf{Q}}(g(e, n)) \\ 0 & \text{otherwise.} \end{cases}$$

This would show that h is recursive if $\mathrm{Prov}_{\mathbf{Q}}$ is. But h is not recursive, by the unsolvability of the Halting Problem (see THM-CMP002, Unsolvability of the Halting Problem, §5.4), so $\mathrm{Prov}_{\mathbf{Q}}$ cannot be either. □

Corollary 6.56 (Undecidability of first-order logic). *First-order logic is undecidable.*

Proof. If first-order logic were decidable, provability in $\mathbf{Q}$ would be as well, since $\mathbf{Q} \vdash \varphi$ iff $\vdash \tau \to \varphi$, where τ is the conjunction of the axioms of $\mathbf{Q}$. □

The previous sections established what first-order logic *cannot* express—limits on decidability, completeness, and definability of specific theories. We now turn to what first-order logic *can* guarantee about the structure of valid implications: Craig's interpolation theorem and Beth's definability theorem reveal a strong modularity property of the entailment relation.

6.9 Craig's Interpolation Theorem

The interpolation theorem states that whenever a valid conditional $\vDash \varphi \to \psi$ holds, there exists a "mediating" sentence χ whose non-logical vocabulary is common to both φ and ψ. Finding such an interpolant amounts to finding a sentence that *separates* φ from $\neg\psi$.

6.9.1 Separation

An interpolant for φ and ψ is a sentence χ such that $\varphi \vDash \chi$ and $\chi \vDash \psi$. By contraposition, the latter holds iff $\neg\psi \vDash \neg\chi$. A sentence χ with this property is said to *separate* φ and $\neg\psi$. So finding an interpolant for φ and ψ amounts to finding a sentence that separates φ and $\neg\psi$. It will be useful to consider the generalization to sets of sentences.

Definition 6.57. A sentence χ *separates* sets of sentences Γ and Δ if and only if $\Gamma \vDash \chi$ and $\Delta \vDash \neg\chi$. If no such sentence exists, then Γ and Δ are *inseparable*.

Example 6.58. Let $\Gamma = \{P(c)\}$ and $\Delta = \{\neg P(c)\}$, where P and c occur in both languages. Then $\chi = P(c)$ separates Γ and Δ: $\Gamma \vDash P(c)$ (trivially) and $\Delta \vDash \neg P(c)$ (trivially). In general, finding an interpolant whose vocabulary is common to both sides is much harder.

Lemma 6.59. *Suppose $\mathcal{L}_0$ is the language containing every constant, function and predicate (other than $\doteq$) that occurs in* both *Γ and Δ, and let $\mathcal{L}_0'$ be obtained by the addition of infinitely many new constants c_n for $n \geq 0$. Then if Γ and Δ are inseparable in $\mathcal{L}_0$, they are also inseparable in $\mathcal{L}_0'$.*

Proof. We proceed indirectly: suppose by way of contradiction that Γ and Δ are separated in $\mathcal{L}_0'$. Then $\Gamma \vDash \chi[c/x]$ and $\Delta \vDash \neg\chi[c/x]$ for some $\chi \in \mathcal{L}_0$ (where c is a new constant—the case where χ contains more than one such new constant is similar). By compactness (Theorem 6.30), there are *finite* subsets Γ_0 of Γ and Δ_0 of Δ such that $\Gamma_0 \vDash \chi[c/x]$ and $\Delta_0 \vDash \neg\chi[c/x]$. Let γ be the conjunction of all formulas in Γ_0 and δ the conjunction of all formulas in Δ_0. Then

$$\gamma \vDash \chi[c/x], \qquad\qquad \delta \vDash \neg\chi[c/x].$$

From the former, by Generalization, we have $\gamma \vDash \forall x\, \chi$, and from the latter by contraposition, $\chi[c/x] \vDash \neg\delta$, whence also $\forall x\, \chi \vDash \neg\delta$. Contraposition again gives $\delta \vDash \neg\forall x\, \chi$. By monotonicity,

$$\Gamma \vDash \forall x\, \chi, \qquad\qquad \Delta \vDash \neg\forall x\, \chi,$$

so that $\forall x\, \chi$ separates Γ and Δ in $\mathcal{L}_0$. □

Lemma 6.60. *Suppose that $\Gamma \cup \{\exists x\, \sigma\}$ and Δ are inseparable, and c is a new constant not in Γ, Δ, or σ. Then $\Gamma \cup \{\exists x\, \sigma, \sigma[c/x]\}$ and Δ are also inseparable.*

Proof. Suppose for contradiction that χ separates $\Gamma \cup \{\exists x\, \sigma, \sigma[c/x]\}$ and Δ, while at the same time $\Gamma \cup \{\exists x \sigma\}$ and Δ are inseparable. We distinguish two cases:

1. c does not occur in χ: in this case $\Gamma \cup \{\exists x\, \sigma, \neg\chi\}$ is satisfiable (otherwise χ separates $\Gamma \cup \{\exists x\, \sigma\}$ and Δ). It remains so if $\sigma[c/x]$ is added, so χ does not separate $\Gamma \cup \{\exists x\, \sigma, \sigma[c/x]\}$ and Δ after all.

2. c does occur in χ so that χ has the form $\chi[c/x]$. Then we have that
$$\Gamma \cup \{\exists x\, \sigma, \sigma[c/x]\} \vDash \chi[c/x],$$
whence $\Gamma, \exists x\, \sigma \vDash \forall x\, (\sigma \to \chi)$ by the Deduction Theorem and Generalization, and finally $\Gamma \cup \{\exists x\, \sigma\} \vDash \exists x\, \chi$. On the other hand, $\Delta \vDash \neg\chi[c/x]$ and hence by Generalization $\Delta \vDash \neg\exists x\, \chi$. So $\Gamma \cup \{\exists x\, \sigma\}$ and Δ are separable, a contradiction.

□

6.9.2 The Interpolation Theorem

Theorem 6.61 (Craig's Interpolation Theorem). *If $\vDash \varphi \to \psi$, then there is a sentence χ such that $\vDash \varphi \to \chi$ and $\vDash \chi \to \psi$, and every constant, function, and predicate (other than $=$) in χ occurs both in φ and ψ. The sentence χ is called an* interpolant *of φ and ψ.*

Proof idea. We prove the contrapositive: if $\{\varphi\}$ and $\{\neg\psi\}$ are inseparable (no sentence in their common vocabulary entails one and refutes the other), then $\varphi \wedge \neg\psi$ is satisfiable, so $\nvDash \varphi \to \psi$. The key step uses Lindenbaum's lemma to extend both sets to maximally consistent sets that agree on the shared vocabulary, then constructs a single structure satisfying both.

Proof. Suppose $\mathcal{L}_1$ is the language of φ and $\mathcal{L}_2$ is the language of ψ. Let $\mathcal{L}_0 = \mathcal{L}_1 \cap \mathcal{L}_2$. For each $i \in \{0, 1, 2\}$, let $\mathcal{L}'_i$ be obtained from $\mathcal{L}_i$ by adding the infinitely many new constants $c_0, c_1, c_2, \ldots$.

If φ is unsatisfiable, $\exists x\, x \neq x$ is an interpolant. If $\neg\psi$ is unsatisfiable (and hence ψ is valid), $\exists x\, x = x$ is an interpolant. So we may assume also that both φ and $\neg\psi$ are satisfiable.

In order to prove the contrapositive of the Interpolation Theorem, assume that there is no interpolant for φ and ψ. In other words, assume that $\{\varphi\}$ and $\{\neg\psi\}$ are inseparable in $\mathcal{L}_0$.

Our goal is to extend the pair $(\{\varphi\}, \{\neg\psi\})$ to a maximally inseparable pair (Γ^*, Δ^*). Let φ_0, φ_1, φ_2, ... enumerate the sentences of $\mathcal{L}_1$, and ψ_0, ψ_1, ψ_2, ... enumerate the sentences of $\mathcal{L}_2$. We define two increasing sequences of sets of sentences (Γ_n, Δ_n), for $n \geq 0$, as follows. Put $\Gamma_0 = \{\varphi\}$ and $\Delta_0 = \{\neg\psi\}$. Assuming (Γ_n, Δ_n) are already defined, define Γ_{n+1} and Δ_{n+1} by:

1. If $\Gamma_n \cup \{\varphi_n\}$ and Δ_n are inseparable in $\mathcal{L}'_0$, put φ_n in Γ_{n+1}. Moreover, if φ_n is an existential formula $\exists x\, \sigma$ then pick a new constant c not occurring in Γ_n, Δ_n, φ_n or ψ_n, and put $\sigma[c/x]$ in Γ_{n+1}.

2. If Γ_{n+1} and $\Delta_n \cup \{\psi_n\}$ are inseparable in $\mathcal{L}'_0$, put ψ_n in Δ_{n+1}. Moreover, if ψ_n is an existential formula $\exists x\, \sigma$, then pick a new constant c not occurring in Γ_{n+1}, Δ_n, φ_n or ψ_n, and put $\sigma[c/x]$ in Δ_{n+1}.

Finally, define:

$$\Gamma^* = \bigcup_{n \geq 0} \Gamma_n, \qquad \Delta^* = \bigcup_{n \geq 0} \Delta_n.$$

By simultaneous induction on n we can now prove:

1. Γ_n and Δ_n are inseparable in $\mathcal{L}'_0$;

2. Γ_{n+1} and Δ_n are inseparable in $\mathcal{L}'_0$.

The basis for 1 is given by Lemma 6.59. For part 2, we need to distinguish three cases:

1. If $\Gamma_0 \cup \{\varphi_0\}$ and Δ_0 are separable, then $\Gamma_1 = \Gamma_0$ and 2 is just 1;

2. If $\Gamma_1 = \Gamma_0 \cup \{\varphi_0\}$, then Γ_1 and Δ_0 are inseparable by construction.

3. It remains to consider the case where φ_0 is existential, so that $\Gamma_1 = \Gamma_0 \cup \{\exists x\, \sigma, \sigma[c/x]\}$. By construction, $\Gamma_0 \cup \{\exists x\, \sigma\}$ and Δ_0 are inseparable, so that by Lemma 6.60 also $\Gamma_0 \cup \{\exists x\, \sigma, \sigma[c/x]\}$ and Δ_0 are inseparable.

This completes the basis of the induction for 1 and 2 above. Now for the inductive step. For 1, if $\Delta_{n+1} = \Delta_n \cup \{\psi_n\}$ then Γ_{n+1} and Δ_{n+1} are inseparable by construction (even when ψ_n is existential, by Lemma 6.60); if $\Delta_{n+1} = \Delta_n$ (because Γ_{n+1} and $\Delta_n \cup \{\psi_n\}$ are separable), then we use the induction hypothesis on 2. For the inductive step for 2, if $\Gamma_{n+2} = \Gamma_{n+1} \cup \{\varphi_{n+1}\}$ then Γ_{n+2} and Δ_{n+1} are inseparable by construction (even when φ_{n+1} is existential, by Lemma 6.60); and if $\Gamma_{n+2} = \Gamma_{n+1}$ then we use the inductive case for 1 just proved. This concludes the induction on 1 and 2.

It follows that Γ^* and Δ^* are inseparable; if not, by compactness, there is $n \geq 0$ that separates Γ_n and Δ_n, against 1. In particular, Γ^* and Δ^* are consistent: for if the former or the latter is inconsistent, then they are separated by $\exists x\, x \neq x$ or $\forall x\, x = x$, respectively.

We now show that Γ^* is maximally consistent in $\mathcal{L}'_1$ and likewise Δ^* in $\mathcal{L}'_2$. For the former, suppose that $\varphi_n \notin \Gamma^*$ and $\neg\varphi_n \notin \Gamma^*$, for some $n \geq 0$. If $\varphi_n \notin \Gamma^*$ then $\Gamma_n \cup \{\varphi_n\}$ is separable from Δ_n, and so there is $\chi \in \mathcal{L}'_0$ such that both:

$$\Gamma^* \vDash \varphi_n \rightarrow \chi, \qquad \Delta^* \vDash \neg\chi.$$

Likewise, if $\neg\varphi_n \notin \Gamma^*$, there is $\chi' \in \mathcal{L}_0'$ such that both:

$$\Gamma^* \vDash \neg\varphi_n \to \chi', \qquad\qquad \Delta^* \vDash \neg\chi'.$$

By propositional logic, $\Gamma^* \vDash \chi \lor \chi'$ and $\Delta^* \vDash \neg(\chi \lor \chi')$, so $\chi \lor \chi'$ separates Γ^* and Δ^*. A similar argument establishes that Δ^* is maximal.

Finally, we show that $\Gamma^* \cap \Delta^*$ is maximally consistent in $\mathcal{L}_0'$. It is obviously consistent, since it is the intersection of consistent sets. To show maximality, let $\sigma \in \mathcal{L}_0'$. Now, Γ^* is maximal in $\mathcal{L}_1' \supseteq \mathcal{L}_0'$, and similarly Δ^* is maximal in $\mathcal{L}_2' \supseteq \mathcal{L}_0'$. It follows that either $\sigma \in \Gamma^*$ or $\neg\sigma \in \Gamma^*$, and either $\sigma \in \Delta^*$ or $\neg\sigma \in \Delta^*$. If $\sigma \in \Gamma^*$ and $\neg\sigma \in \Delta^*$ then σ would separate Γ^* and Δ^*; and if $\neg\sigma \in \Gamma^*$ and $\sigma \in \Delta^*$ then Γ^* and Δ^* would be separated by $\neg\sigma$. Hence, either $\sigma \in \Gamma^* \cap \Delta^*$ or $\neg\sigma \in \Gamma^* \cap \Delta^*$, and $\Gamma^* \cap \Delta^*$ is maximal.

Since Γ^* is maximally consistent, it has a model $\mathfrak{M}_1'$ whose domain $|\mathfrak{M}_1'|$ comprises all and only the elements $c^{\mathfrak{M}_1'}$ interpreting the constants—just like in the proof of the Completeness Theorem (Theorem 6.27). Similarly, Δ^* has a model $\mathfrak{M}_2'$ whose domain $|\mathfrak{M}_2'|$ is given by the interpretations $c^{\mathfrak{M}_2'}$ of the constants.

Let $\mathfrak{M}_1$ be obtained from $\mathfrak{M}_1'$ by dropping interpretations for constants, functions, and predicates in $\mathcal{L}_1' \setminus \mathcal{L}_0'$, and similarly for $\mathfrak{M}_2$. Then the map $h\colon M_1 \to M_2$ defined by $h(c^{\mathfrak{M}_1'}) = c^{\mathfrak{M}_2'}$ is an isomorphism in $\mathcal{L}_0'$, because $\Gamma^* \cap \Delta^*$ is maximally consistent in $\mathcal{L}_0'$, as shown. This follows because any $\mathcal{L}_0'$-sentence either belongs to both Γ^* and Δ^*, or to neither: so $c^{\mathfrak{M}_1'} \in P^{\mathfrak{M}_1'}$ if and only if $P(c) \in \Gamma^*$ if and only if $P(c) \in \Delta^*$ if and only if $c^{\mathfrak{M}_2'} \in P^{\mathfrak{M}_2'}$. The other conditions satisfied by isomorphisms can be established similarly.

Let us now define a model $\mathfrak{M}$ for the language $\mathcal{L}_1 \cup \mathcal{L}_2$ as follows:

1. The domain $|\mathfrak{M}|$ is just $|\mathfrak{M}_2|$, i.e., the set of all elements $c^{\mathfrak{M}_2'}$;

2. If a predicate P is in $\mathcal{L}_2 \setminus \mathcal{L}_1$ then $P^{\mathfrak{M}} = P^{\mathfrak{M}_2'}$;

3. If a predicate P is in $\mathcal{L}_1 \setminus \mathcal{L}_2$ then $P^{\mathfrak{M}} = h(P^{\mathfrak{M}_2'})$, i.e., $\langle c_1^{\mathfrak{M}_2'}, \dots, c_n^{\mathfrak{M}_2'}\rangle \in P^{\mathfrak{M}}$ if and only if $\langle c_1^{\mathfrak{M}_1'}, \dots, c_n^{\mathfrak{M}_1'}\rangle \in P^{\mathfrak{M}_1'}$.

4. If a predicate P is in $\mathcal{L}_0$ then $P^{\mathfrak{M}} = P^{\mathfrak{M}_2'} = h(P^{\mathfrak{M}_1'})$.

5. Functions of $\mathcal{L}_1 \cup \mathcal{L}_2$, including constants, are handled similarly.

Finally, one shows by induction on formulas that $\mathfrak{M}$ agrees with $\mathfrak{M}_1'$ on all formulas of $\mathcal{L}_1'$ and with $\mathfrak{M}_2'$ on all formulas of $\mathcal{L}_2'$. In particular, $\mathfrak{M} \vDash \Gamma^* \cup \Delta^*$, whence $\mathfrak{M} \vDash \varphi$ and $\mathfrak{M} \vDash \neg\psi$, and $\nvDash \varphi \to \psi$. This concludes the proof of Craig's Interpolation Theorem. □

6.10 Beth's Definability Theorem

One important application of the interpolation theorem is Beth's definability theorem. To define an n-place relation R we can give a formula χ with n free variables which does not involve R. This would be an *explicit* definition of R in terms of χ. We can then say also that a theory $\Sigma(P)$ in a language containing the n-place predicate P explicitly defines P if it contains (or at least entails) a formalized explicit definition, i.e.,

$$\Sigma(P) \vDash \forall x_1 \dots \forall x_n\, (P(x_1, \dots, x_n) \leftrightarrow \chi(x_1, \dots, x_n)).$$

But an explicit definition is only one way of defining—in the sense of determining completely—a relation. A theory may also be such that the interpretation of P is fixed by the interpretation of the rest of the language in any model. The definability theorem states that whenever a theory fixes the interpretation of P in this way—whenever it *implicitly defines* P—then it also explicitly defines it.

Definition 6.62. Suppose $\mathcal{L}$ is a language not containing the predicate P. A set $\Sigma(P)$ of sentences of $\mathcal{L} \cup \{P\}$ *explicitly defines* P if and only if there is a formula $\chi(x_1, \dots, x_n)$ of $\mathcal{L}$ such that

$$\Sigma(P) \vDash \forall x_1 \dots \forall x_n\, (P(x_1, \dots, x_n) \leftrightarrow \chi(x_1, \dots, x_n)).$$

The distinction matters: explicit definability gives a concrete formula χ equivalent to P; implicit definability merely guarantees uniqueness—any two predicates satisfying the same axioms must be identical. Beth's theorem (below) shows that, remarkably, these two notions coincide in first-order logic.

Definition 6.63. Suppose $\mathcal{L}$ is a language not containing the predicates P and P'. A set $\Sigma(P)$ of sentences of $\mathcal{L} \cup \{P\}$ *implicitly defines* P if and only if

$$\Sigma(P) \cup \Sigma(P') \vDash \forall x_1 \dots \forall x_n\, (P(x_1, \dots, x_n) \leftrightarrow P'(x_1, \dots, x_n)),$$

where $\Sigma(P')$ is the result of uniformly replacing P with P' in $\Sigma(P)$.

In other words, for any model $\mathfrak{M}$ and $R, R' \subseteq |\mathfrak{M}|^n$, if both $(\mathfrak{M}, R) \vDash \Sigma(P)$ and $(\mathfrak{M}, R') \vDash \Sigma(P')$, then $R = R'$; where $(\mathfrak{M}, R)$ is the structure $\mathfrak{M}'$ for the expansion of $\mathcal{L}$ to $\mathcal{L} \cup \{P\}$ such that $P^{\mathfrak{M}'} = R$, and similarly for $(\mathfrak{M}, R')$.

Theorem 6.64 (Beth Definability Theorem). *A set $\Sigma(P)$ of $\mathcal{L} \cup \{P\}$-formulas implicitly defines P if and only $\Sigma(P)$ explicitly defines P.*

Proof. If $\Sigma(P)$ explicitly defines P then both

$$\begin{aligned}\Sigma(P) &\vDash \qquad \forall x_1 \ldots \forall x_n\, (P(x_1, \ldots, x_n) \leftrightarrow \chi(x_1, \ldots, x_n))\\ \Sigma(P') &\vDash \qquad \forall x_1 \ldots \forall x_n\, (P'(x_1, \ldots, x_n) \leftrightarrow \chi(x_1, \ldots, x_n))\end{aligned}$$

and the conclusion follows. For the converse: assume that $\Sigma(P)$ implicitly defines P. First, we add constants $c_1, \ldots, c_n$ to $\mathcal{L}$. Then

$$\Sigma(P) \cup \Sigma(P') \vDash P(c_1, \ldots, c_n) \rightarrow P'(c_1, \ldots, c_n).$$

By compactness (Theorem 6.30), there are finite sets $\Delta_0 \subseteq \Sigma(P)$ and $\Delta_1 \subseteq \Sigma(P')$ such that

$$\Delta_0 \cup \Delta_1 \vDash P(c_1, \ldots, c_n) \rightarrow P'(c_1, \ldots, c_n).$$

Let $\theta(P)$ be the conjunction of all sentences $\varphi(P)$ such that either $\varphi(P) \in \Delta_0$ or $\varphi(P') \in \Delta_1$ and let $\theta(P')$ be the conjunction of all sentences $\varphi(P')$ such that either $\varphi(P) \in \Delta_0$ or $\varphi(P') \in \Delta_1$. Then $\theta(P) \land \theta(P') \vDash P(c_1, \ldots, c_n) \rightarrow P'c_1 \ldots c_n$. We can re-arrange this so that each predicate occurs on one side of $\vDash$:

$$\theta(P) \land P(c_1, \ldots, c_n) \vDash \theta(P') \rightarrow P'(c_1, \ldots, c_n).$$

By Craig's Interpolation Theorem (Theorem 6.61) there is a sentence $\chi(c_1, \ldots, c_n)$ not containing P or P' such that:

$$\begin{aligned}\theta(P) \land P(c_1, \ldots, c_n) &\vDash \chi(c_1, \ldots, c_n);\\ \chi(c_1, \ldots, c_n) &\vDash \theta(P') \rightarrow P'(c_1, \ldots, c_n).\end{aligned}$$

From the former of these two entailments we have: $\theta(P) \vDash P(c_1, \ldots, c_n) \rightarrow \chi(c_1, \ldots, c_n)$. And from the latter, since an $\mathcal{L} \cup \{P\}$-model $(\mathfrak{M}, R) \vDash \varphi(P)$ if and only if the corresponding $\mathcal{L} \cup \{P'\}$-model $(\mathfrak{M}, R) \models \varphi(P')$, we have $\chi(c_1, \ldots, c_n) \vDash \theta(P) \rightarrow P(c_1, \ldots, c_n)$, from which:

$$\theta(P) \vDash \chi(c_1, \ldots, c_n) \rightarrow P(c_1, \ldots, c_n).$$

Putting the two together, $\theta(P) \vDash P(c_1, \ldots, c_n) \leftrightarrow \chi(c_1, \ldots, c_n)$, and by monotonicity and generalization also

$$\Sigma(P) \vDash \forall x_1 \ldots \forall x_n\, (P(x_1, \ldots, x_n) \leftrightarrow \chi(x_1, \ldots, x_n)).$$

□

6.11 Lindström's Theorem

Lindström's theorem characterizes first-order logic as the maximal logic—in a precisely defined sense—for which both the Compactness Theorem and the Downward Löwenheim–Skolem Theorem hold. To state the theorem we need the notions of abstract logic, partial isomorphism, and the back-and-forth characterization of n-equivalence. Throughout this section we restrict to purely relational languages (containing only predicates and individual constants, no functions).

6.11.1 Abstract Logics

Definition 6.65. An *abstract logic* is a pair $\langle L, \models_L \rangle$, where L is a function that assigns to each language $\mathcal{L}$ a set $L(\mathcal{L})$ of sentences, and $\models_L$ is a relation between structures for the language $\mathcal{L}$ and elements of $L(\mathcal{L})$. In particular, $\langle F, \models \rangle$ is ordinary first-order logic, i.e., F is the function assigning to the language $\mathcal{L}$ the set of first-order sentences built from the constants in $\mathcal{L}$, and $\models$ is the satisfaction relation of first-order logic.

Definition 6.66. Let $\mathrm{Mod}_L(\alpha)$ denote the class $\{\mathfrak{M} : \mathfrak{M} \models_L \alpha\}$. If the language needs to be made explicit, we write $\mathrm{Mod}_L^{\mathcal{L}}(\alpha)$. Two structures $\mathfrak{M}$ and $\mathfrak{N}$ for $\mathcal{L}$ are *elementarily equivalent in* $\langle L, \models_L \rangle$, written $\mathfrak{M} \equiv_L \mathfrak{N}$, if the same sentences from $L(\mathcal{L})$ are true in each.

Definition 6.67. An abstract logic $\langle L, \models_L \rangle$ for the language $\mathcal{L}$ is *normal* if it satisfies the following properties:

1. (*L-Monotonicity*) For languages $\mathcal{L}$ and $\mathcal{L}'$, if $\mathcal{L} \subseteq \mathcal{L}'$, then $L(\mathcal{L}) \subseteq L(\mathcal{L}')$.
2. (*Expansion Property*) For each $\alpha \in L(\mathcal{L})$ there is a *finite* subset $\mathcal{L}'$ of $\mathcal{L}$ such that the relation $\mathfrak{M} \models_L \alpha$ depends only on the reduct of $\mathfrak{M}$ to $\mathcal{L}'$.
3. (*Isomorphism Property*) If $\mathfrak{M} \models_L \alpha$ and $\mathfrak{M} \simeq \mathfrak{N}$ then also $\mathfrak{N} \models_L \alpha$.
4. (*Renaming Property*) The relation $\models_L$ is preserved under renaming of non-logical symbols.
5. (*Boolean Property*) $\langle L, \models_L \rangle$ is closed under the Boolean connectives: for each $\alpha \in L(\mathcal{L})$ there is β with $\mathrm{Mod}_L(\beta) = \mathrm{Mod}_L(\alpha)^c$, and for each pair α, β there is γ with $\mathrm{Mod}_L(\gamma) = \mathrm{Mod}_L(\alpha) \cap \mathrm{Mod}_L(\beta)$.
6. (*Quantifier Property*) For each constant c in $\mathcal{L}$ and $\alpha \in L(\mathcal{L})$ there is $\beta \in L(\mathcal{L} \setminus \{c\})$ such that $\mathfrak{M} \models_L \beta$ iff $(\mathfrak{M}, a) \models_L \alpha$ for some $a \in |\mathfrak{M}|$.
7. (*Relativization Property*) Given a sentence $\alpha \in L(\mathcal{L})$ and symbols R, c_1, $\dots$, c_n not in $\mathcal{L}$, there is a sentence β (the *relativization* of α) such that satisfaction of β in an expansion of $\mathfrak{M}$ corresponds to satisfaction of α in the substructure picked out by R.

First-order logic $\langle F, \models \rangle$ is normal. Moreover, if $\langle L, \models_L \rangle$ is normal, then $\langle F, \models \rangle \leq \langle L, \models_L \rangle$.

Definition 6.68. Given two abstract logics $\langle L_1, \models_{L_1}\rangle$ and $\langle L_2, \models_{L_2}\rangle$ we say that the latter is *at least as expressive* as the former, written $\langle L_1, \models_{L_1}\rangle \leq \langle L_2, \models_{L_2}\rangle$, if for each language $\mathcal{L}$ and sentence $\alpha \in L_1(\mathcal{L})$ there is a sentence $\beta \in L_2(\mathcal{L})$ such that $\mathrm{Mod}^{\mathcal{L}}_{L_1}(\alpha) = \mathrm{Mod}^{\mathcal{L}}_{L_2}(\beta)$. The logics are *equivalent* if the inequality holds in both directions.

6.11.2 Compactness and Löwenheim–Skolem Properties

Definition 6.69. An abstract logic $\langle L, \models_L\rangle$ has the *Compactness Property* if each set Γ of $L(\mathcal{L})$-sentences is satisfiable whenever each finite $\Gamma_0 \subseteq \Gamma$ is satisfiable.

Definition 6.70. $\langle L, \models_L\rangle$ has the *Downward Löwenheim–Skolem Property* if any satisfiable Γ has an enumerable model.

6.11.3 Partial Isomorphisms

Definition 6.71. Given two structures $\mathfrak{M}$ and $\mathfrak{N}$, a *partial isomorphism* from $\mathfrak{M}$ to $\mathfrak{N}$ is a finite partial function p taking arguments in $|\mathfrak{M}|$ and returning values in $|\mathfrak{N}|$, which is injective and preserves the interpretations of all constants, predicates, and functions on its domain.

Definition 6.72. Two structures $\mathfrak{M}$ and $\mathfrak{N}$ are *partially isomorphic*, written $\mathfrak{M} \simeq_p \mathfrak{N}$, if and only if there is a non-empty set I of partial isomorphisms between $\mathfrak{M}$ and $\mathfrak{N}$ satisfying the *back-and-forth* property:

1. (*Forth*) For every $p \in I$ and $a \in |\mathfrak{M}|$ there is $q \in I$ such that $p \subseteq q$ and a is in the domain of q;
2. (*Back*) For every $p \in I$ and $b \in |\mathfrak{N}|$ there is $q \in I$ such that $p \subseteq q$ and b is in the range of q.

Theorem 6.73. *If $\mathfrak{M} \simeq_p \mathfrak{N}$ and $\mathfrak{M}$ and $\mathfrak{N}$ are enumerable, then $\mathfrak{M} \simeq \mathfrak{N}$.*

Proof sketch. Enumerate $|\mathfrak{M}| = \{a_0, a_1, \dots\}$ and $|\mathfrak{N}| = \{b_0, b_1, \dots\}$. Starting from an arbitrary $p_0 \in I$, alternately apply the Forth property (to include a_r in the domain) and the Back property (to include b_r in the range), building an increasing chain $p_0 \subseteq p_1 \subseteq \cdots$. The union $p = \bigcup_n p_n$ is an isomorphism. □

Theorem 6.74. *Suppose $\mathfrak{M}$ and $\mathfrak{N}$ are structures for a purely relational language. Then if $\mathfrak{M} \simeq_p \mathfrak{N}$, also $\mathfrak{M} \equiv \mathfrak{N}$.*

Proof sketch. By induction on formulas, one shows that if p maps each a_i to b_i, then $\mathfrak{M}, s_1 \vDash \varphi$ iff $\mathfrak{N}, s_2 \vDash \varphi$ whenever $s_1(x_i) = a_i$ and $s_2(x_i) = b_i$. The base case uses the isomorphism conditions on p; the quantifier step uses the back-and-forth property. The case $n = 0$ gives $\mathfrak{M} \equiv \mathfrak{N}$. □

6.11.4 Quantifier Rank and n-Equivalence

Definition 6.75. For any formula φ, the *quantifier rank* of φ, denoted by $\mathrm{qr}(\varphi) \in \mathbb{N}$, is recursively defined as the highest number of nested quantifiers in φ. Two structures $\mathfrak{M}$ and $\mathfrak{N}$ are *n-equivalent*, written $\mathfrak{M} \equiv_n \mathfrak{N}$, if they agree on all sentences of quantifier rank less than or equal to n.

Proposition 6.76. *Let $\mathcal{L}$ be a finite purely relational language. Then for each $n \in \mathbb{N}$ there are only finitely many first-order sentences in $\mathcal{L}$ that have quantifier rank no greater than n, up to logical equivalence.*

Proof. By induction on n. □

Definition 6.77. Given structures $\mathfrak{M}$ and $\mathfrak{N}$, we define relations $I_n \subseteq |\mathfrak{M}|^{<\omega} \times |\mathfrak{N}|^{<\omega}$ between sequences of equal length, by recursion on n as follows:

1. $I_0(\mathbf{a}, \mathbf{b})$ iff $\mathbf{a}$ and $\mathbf{b}$ satisfy the same atomic formulas in $\mathfrak{M}$ and $\mathfrak{N}$.
2. $I_{n+1}(\mathbf{a}, \mathbf{b})$ iff for every $a \in |\mathfrak{M}|$ there is a $b \in |\mathfrak{N}|$ such that $I_n(\mathbf{a}a, \mathbf{b}b)$, and vice-versa.

Definition 6.78. Write $\mathfrak{M} \approx_n \mathfrak{N}$ if $I_n(\Lambda, \Lambda)$ holds of $\mathfrak{M}$ and $\mathfrak{N}$ (where Λ is the empty sequence).

Theorem 6.79. *Let $\mathcal{L}$ be a purely relational language. Then $I_n(\mathbf{a}, \mathbf{b})$ implies that for every φ such that $\mathrm{qr}(\varphi) \leq n$, we have $\mathfrak{M}, \mathbf{a} \vDash \varphi$ if and only if $\mathfrak{N}, \mathbf{b} \vDash \varphi$. Moreover, if $\mathcal{L}$ is finite, the converse also holds.*

Proof sketch. The forward direction proceeds by induction on φ. For the converse, one proceeds by induction on n. The key step uses Proposition 6.76: given $a \in |\mathfrak{M}|$, let τ_n^a be the finite set of formulas of rank $\leq n$ satisfied by $\mathbf{a}a$ in $\mathfrak{M}$. Then $\mathbf{a}$ satisfies $\exists x\, \tau_n^a$ (rank $\leq n+1$), so by hypothesis $\mathbf{b}$ does too in $\mathfrak{N}$, yielding the required b. □

Corollary 6.80. *If $\mathfrak{M}$ and $\mathfrak{N}$ are purely relational structures in a finite language, then $\mathfrak{M} \approx_n \mathfrak{N}$ if and only if $\mathfrak{M} \equiv_n \mathfrak{N}$. In particular $\mathfrak{M} \equiv \mathfrak{N}$ if and only if for each n, $\mathfrak{M} \approx_n \mathfrak{N}$.*

6.11.5 Partially Isomorphic Structures in Abstract Logics

The notion of partial isomorphism is purely algebraic and hence applies to abstract logics. The following key theorem shows that if $\langle L, \models_L \rangle$ has the Löwenheim–Skolem property, partially isomorphic structures are L-equivalent.

Theorem 6.81. *Suppose $\langle L, \models_L \rangle$ is a normal logic with the Löwenheim–Skolem property. Then any two structures that are partially isomorphic are elementarily equivalent in $\langle L, \models_L \rangle$.*

Proof sketch. Suppose $\mathfrak{M} \simeq_p \mathfrak{N}$ but $\mathfrak{M} \models_L \alpha$ while $\mathfrak{N} \not\models_L \alpha$. Using the Isomorphism and Expansion Properties, assume $|\mathfrak{M}|$ and $|\mathfrak{N}|$ are disjoint and $\alpha \in L(\mathcal{L})$ for a finite language. Encode the partial isomorphism I and the extended structures $\mathfrak{M}^*$, $\mathfrak{N}^*$ (with their finite-sequence domains and concatenation predicates) into a single structure $\mathfrak{M}$. The Relativization Property yields a first-order sentence θ_1 true in $\mathfrak{M}$ expressing that $\mathfrak{M} \models_L \alpha$ and $\mathfrak{N} \not\models_L \alpha$, and a sentence θ_2 expressing that $\mathfrak{M} \simeq_p \mathfrak{N}$ via I. By the Löwenheim–Skolem Property, $\theta_1 \wedge \theta_2$ has an enumerable model containing enumerable partially isomorphic substructures. But enumerable partially isomorphic structures are isomorphic (Theorem 6.73), contradicting the Isomorphism Property. □

6.11.6 Lindström's Theorem

Lemma 6.82. *Suppose $\alpha \in L(\mathcal{L})$, with $\mathcal{L}$ finite, and assume also that there is an $n \in \mathbb{N}$ such that for any two structures $\mathfrak{M}$ and $\mathfrak{N}$, if $\mathfrak{M} \equiv_n \mathfrak{N}$ and $\mathfrak{M} \models_L \alpha$ then also $\mathfrak{N} \models_L \alpha$. Then α is equivalent to a first-order sentence, i.e., there is a first-order θ such that* $\mathrm{Mod}_L(\alpha) = \mathrm{Mod}_L(\theta)$.

Proof. Let n be such that any two n-equivalent structures $\mathfrak{M}$ and $\mathfrak{N}$ agree on the value assigned to α. Recall Proposition 6.76: there are only finitely many first-order sentences in a finite language that have quantifier rank no greater than n, up to logical equivalence. Now, for each fixed structure $\mathfrak{M}$ let $\theta_{\mathfrak{M}}$ be the conjunction of all first-order sentences α true in $\mathfrak{M}$ with $\mathrm{qr}(\alpha) \leq n$ (this conjunction is finite), so that $\mathfrak{N} \models \theta_{\mathfrak{M}}$ if and only if $\mathfrak{N} \equiv_n \mathfrak{M}$. Then put $\theta = \bigvee\{\theta_{\mathfrak{M}} : \mathfrak{M} \models_L \alpha\}$; this disjunction is also finite (up to logical equivalence).

The conclusion $\mathrm{Mod}_L(\alpha) = \mathrm{Mod}_L(\theta)$ follows. In fact, if $\mathfrak{N} \models_L \theta$ then for some $\mathfrak{M} \models_L \alpha$ we have $\mathfrak{N} \models \theta_{\mathfrak{M}}$, whence also $\mathfrak{N} \models_L \alpha$ (by the hypothesis of the lemma). Conversely, if $\mathfrak{N} \models_L \alpha$ then $\theta_{\mathfrak{N}}$ is a disjunct in θ, and since $\mathfrak{N} \models \theta_{\mathfrak{N}}$, also $\mathfrak{N} \models_L \theta$. □

Theorem 6.83 (Lindström's Theorem)**.** *Suppose $\langle L, \models_L\rangle$ has the Compactness and the Löwenheim–Skolem Properties. Then $\langle L, \models_L\rangle \leq \langle F, \models\rangle$ (so $\langle L, \models_L\rangle$ is equivalent to first-order logic).*

Proof idea. Given an L-sentence α over a finite relational language, use the back-and-forth characterization of elementary equivalence (Theorem 6.79) and a compactness argument to find n such that α cannot distinguish structures agreeing on all first-order sentences of quantifier rank $\leq n$. Then α is equivalent to a Boolean combination of such sentences, showing $\langle L, \models_L\rangle \leq \langle F, \models\rangle$.

Proof. By Lemma 6.82, it suffices to show that for any $\alpha \in L(\mathcal{L})$, with $\mathcal{L}$ finite, there is $n \in \mathbb{N}$ such that for any two structures $\mathfrak{M}$ and $\mathfrak{N}$: if $\mathfrak{M} \equiv_n \mathfrak{N}$ then $\mathfrak{M}$ and $\mathfrak{N}$ agree on α. For then α is equivalent to a first-order sentence, from which $\langle L, \models_L\rangle \leq \langle F, \models\rangle$ follows. Since we are working in a finite, purely relational language, by Theorem 6.79 we can replace the statement that $\mathfrak{M} \equiv_n \mathfrak{N}$ by the corresponding algebraic statement that $I_n(\emptyset, \emptyset)$.

Given α, suppose towards a contradiction that for each n there are structures $\mathfrak{M}_n$ and $\mathfrak{N}_n$ such that $I_n(\emptyset, \emptyset)$, but (say) $\mathfrak{M}_n \models_L \alpha$ whereas $\mathfrak{N}_n \not\models_L \alpha$. By the Isomorphism Property we can assume that all the $\mathfrak{M}_n$'s interpret the constants of the language by the same objects; furthermore, since there are only finitely many atomic sentences in the language, we may also assume that they satisfy the same atomic sentences (we can take a subsequence of the $\mathfrak{M}$'s otherwise). Let $\mathfrak{M}$ be the union of all the $\mathfrak{M}_n$'s, i.e., the unique minimal structure having each $\mathfrak{M}_n$ as a substructure. As in the proof of Theorem 6.81, let $\mathfrak{M}^*$ be the extension of $\mathfrak{M}$ with domain $|\mathfrak{M}| \cup |\mathfrak{M}|^{<\omega}$, in the expanded language comprising the concatenation predicates P and Q.

Similarly, define $\mathfrak{N}_n$, $\mathfrak{N}$ and $\mathfrak{N}^*$. Now let $\mathfrak{M}$ be the structure whose domain comprises the domains of $\mathfrak{M}^*$ and $\mathfrak{N}^*$ as well as the natural numbers $\mathbb{N}$ along with their natural ordering $\leq$, in the language with extra predicates representing the domains $|\mathfrak{M}|$, $|\mathfrak{N}|$, $|\mathfrak{M}|^{<\omega}$ and $|\mathfrak{N}|^{<\omega}$ as well as predicates coding the domains of $\mathfrak{M}_n$ and $\mathfrak{N}_n$ in the sense that:

$$|\mathfrak{M}_n| = \{a \in |\mathfrak{M}| : R(a, n)\}; \qquad |\mathfrak{N}_n| = \{a \in |\mathfrak{N}| : S(a, n)\};$$
$$|\mathfrak{M}|^{<\omega}_n = \{a \in |\mathfrak{M}|^{<\omega} : R(a, n)\}; \qquad |\mathfrak{N}|^{<\omega}_n = \{a \in |\mathfrak{N}|^{<\omega} : S(a, n)\}.$$

The structure $\mathfrak{M}$ also has a ternary relation J such that $J(n, \mathbf{a}, \mathbf{b})$ holds if and only if $I_n(\mathbf{a}, \mathbf{b})$.

Now there is a sentence θ in the language $\mathcal{L}$ augmented by R, S, J, etc., saying that $\leq$ is a discrete linear ordering with first but no last element and such that $\mathfrak{M}_n \models \alpha$, $\mathfrak{N}_n \not\models \alpha$, and for each n in the ordering, $J(n, \mathbf{a}, \mathbf{b})$ holds if and only if $I_n(\mathbf{a}, \mathbf{b})$.

Using the Compactness Property, we can find a model $\mathfrak{M}^*$ of θ in which the ordering contains a non-standard element n^*. In particular then $\mathfrak{M}^*$ will contain substructures $\mathfrak{M}_{n^*}$ and $\mathfrak{N}_{n^*}$ such that $\mathfrak{M}_{n^*} \models_L \alpha$ and $\mathfrak{N}_{n^*} \not\models_L \alpha$. But now we can define a set $\mathcal{I}$ of pairs of k-tuples from $|\mathfrak{M}_{n^*}|$ and $|\mathfrak{N}_{n^*}|$ by putting $\langle \mathbf{a}, \mathbf{b} \rangle \in \mathcal{I}$ if and only if $J(n^* - k, \mathbf{a}, \mathbf{b})$, where k is the length of $\mathbf{a}$ and $\mathbf{b}$. Since n^* is non-standard, for each standard k we have that $n^* - k > 0$, and the set $\mathcal{I}$ witnesses the fact that $\mathfrak{M}_{n^*} \simeq_p \mathfrak{N}_{n^*}$. But by Theorem 6.81, $\mathfrak{M}_{n^*}$ is L-equivalent to $\mathfrak{N}_{n^*}$, a contradiction. □

6.12 Equivalence of Proof Systems

We have introduced four architectures for deriving theorems of classical first-order logic: axiomatic (Hilbert-style) deduction (see DEF-DED005, §4.2), natural deduction (see DEF-DED006, §4.3), the sequent calculus (see DEF-DED007, §4.4), and analytic tableaux (see DEF-DED008, §4.5). Despite their very different structures, all four systems derive exactly the same formulas.

Theorem 6.84 (Equivalence of Proof Systems). *For classical first-order logic, the following are equivalent for any set of sentences Γ and sentence φ:*

1. $\Gamma \vdash_H \varphi$ *(derivable in the Hilbert calculus);*
2. $\Gamma \vdash_{ND} \varphi$ *(derivable in natural deduction);*
3. *The sequent $\Gamma \Rightarrow \varphi$ is derivable in the sequent calculus;*
4. *The tableau for $\Gamma \cup \{\neg\varphi\}$ closes.*

All four proof system architectures derive exactly the same formulas.

Proof sketch. One establishes a cycle of mutual simulations.

Hilbert ⇒ Natural Deduction. Every axiom schema of the Hilbert calculus is derivable in natural deduction (using introduction rules to construct proofs of the corresponding tautological schemas). Modus ponens corresponds to an application of $\to$-elimination. Hence any Hilbert derivation can be transformed step-by-step into a natural deduction derivation.

Natural Deduction ⇒ Sequent Calculus. Each introduction rule of natural deduction corresponds to a right rule of the sequent calculus, and each elimination rule corresponds to a left rule followed by a cut. Discharged assumptions in natural deduction become formulas on the left side of the sequent. The translation proceeds by induction on the structure of the natural deduction derivation tree.

Sequent Calculus $\Rightarrow$ Tableaux. A sequent derivation can be read "upside down" as a closed tableau. An initial sequent $\varphi \Rightarrow \varphi$ corresponds to a branch containing both φ and $\neg\varphi$, and hence closed. Left and right rules of the sequent calculus correspond to the tableau expansion rules for signed formulas. Branching in the sequent calculus (e.g., $\vee$-right, $\wedge$-left) corresponds to branching in the tableau.

Tableaux $\Rightarrow$ Hilbert. A closed tableau for $\Gamma \cup \{\neg\varphi\}$ witnesses the unsatisfiability of $\Gamma \cup \{\neg\varphi\}$. By soundness of tableaux (see §4.5) we have $\Gamma \vDash \varphi$, and by the Completeness Theorem (Theorem 6.27), $\Gamma \vdash_H \varphi$.

Alternatively, each direction can be verified by a direct syntactic translation. The indirect route via soundness and completeness gives the result with less effort: since each system is sound and complete with respect to the same semantics, they derive the same formulas. □

The indirect argument deserves emphasis. By the Soundness Theorem (Theorem 6.1), for each system S we have: if $\Gamma \vdash_S \varphi$ then $\Gamma \vDash \varphi$. By the Completeness Theorem (Theorem 6.27), for each system S: if $\Gamma \vDash \varphi$ then $\Gamma \vdash_S \varphi$. Chaining these two facts for any pair of systems S_1 and S_2 yields: $\Gamma \vdash_{S_1} \varphi$ iff $\Gamma \vdash_{S_2} \varphi$. Thus the equivalence is an immediate corollary of soundness and completeness, and does not require an explicit syntactic simulation—though such simulations are of independent interest for understanding the computational relationships between proof systems.

Chapter 7

Formal Set Theory

7.1 The Language of Set Theory

In Chapter 1 we developed set-theoretic foundations informally, treating sets as intuitively given collections of objects. We now turn to *formal* set theory: a first-order theory in which every concept is defined from a single binary relation symbol, and every existence claim is justified by explicit axioms.

Definition 7.1 (Set (Formal))**.** In ZFC, the variables of $\mathcal{L}_\in$ range over *sets*. Unlike the naive sets of Chapter 1, formal sets are objects within a first-order theory: they exist only to the extent that the ZFC axioms guarantee their existence.

The formal language of set theory is remarkably austere:

Definition 7.2 (Membership)**.** The language $\mathcal{L}_\in = \{\in\}$ has a single binary relation symbol $\in$. All set-theoretic concepts—subset, union, power set, ordinal, cardinal, function—are defined in terms of $\in$ and the logical connectives of first-order logic (see SYN.1–SYN.2, Chapter 2).

The notation $(\forall x \in A)\varphi$ abbreviates $\forall x(x \in A \to \varphi)$, and $(\exists x \in A)\varphi$ abbreviates $\exists x(x \in A \wedge \varphi)$.

Remark 33. It is sometimes useful to speak of *classes*—collections that may be "too large" to be sets (for instance, the collection of all ordinals). Proper classes will be introduced in SET.3, where the Burali-Forti paradox motivates the distinction between sets and proper classes.

7.2 ZFC Axioms

The axioms of ZFC are motivated by the *cumulative-iterative conception* of set, which rests on three informal principles:

> *Stages-are-key*. Every set is formed at some stage.
>
> *Stages-are-ordered*. Stages are ordered: some come *before* others.
>
> *Stages-accumulate*. For any stage S, and for any sets which were formed *before* stage S: a set is formed at stage S whose members are exactly those sets. Nothing else is formed at stage S.

These informal principles do not constitute a formal theory, but they serve to justify each axiom we adopt. We will introduce additional stage-theoretic principles as needed.

7.2.1 Extensionality

The very first thing to say is that sets are individuated by their elements. More precisely:

Axiom (Extensionality)**.** If sets A and B have the same elements, then A and B are the same set.

$$\forall A \, \forall B \, (\forall x \, (x \in A \leftrightarrow x \in B) \rightarrow A = B)$$

The Axiom of Extensionality expresses the basic idea that a set is determined by its elements. (So sets might be contrasted with *concepts*, where precisely the same objects might fall under many different concepts.)

Why embrace this principle? Well, it is plausible to say that any denial of Extensionality is a decision to abandon anything which might even be called *set theory*. Set theory is no more nor less than the theory of extensional collections.

The real challenge, though, is to lay down principles which tell us *which sets exist*. And it turns out that the only truly "obvious" answer to this question is provably wrong.

7.2.2 Separation

We start with a principle to replace Naive Comprehension:

Axiom (Scheme of Separation)**.** For every formula $\varphi(x)$, this is an axiom: for any A, the set $\{x \in A : \varphi(x)\}$ exists.

Note that this is not a single axiom. It is a *scheme* of axioms. There are *infinitely many* Separation axioms; one for every formula $\varphi(x)$. The scheme can equally well be (and normally is) written down as follows:

For any formula $\varphi(x)$ which does not contain "S", this is an axiom:

$$\forall A \exists S \forall x(x \in S \leftrightarrow (\varphi(x) \land x \in A)).$$

The formulas φ in the Separation axioms may have parameters.

Separation is immediately justified by the cumulative-iterative conception. To see why, let A be a set. So A is formed by some stage S (by *Stages-are-key*). Since A was formed at stage S, all of A's members were formed before stage S (by *Stages-accumulate*). Now in particular, consider all the sets which are members of A and which also satisfy φ; clearly all of these sets, too, were formed before stage S. So they are formed into a set $\{x \in A : \varphi(x)\}$ at stage S too (by *Stages-accumulate*).

Unlike Naive Comprehension, this avoids Russell's Paradox. For we cannot simply assert the existence of the set $\{x : x \notin x\}$. Rather, *given* some set A, we can assert the existence of the set $R_A = \{x \in A : x \notin x\}$. But all this proves is that $R_A \notin R_A$ and $R_A \notin A$, none of which is very worrying.

However, Separation has an immediate and striking consequence: there is no *universal* set, i.e., $\{x : x = x\}$ does not exist. For if V were a universal set, then by Separation, $R = \{x \in V : x \notin x\} = \{x : x \notin x\}$ would exist, contradicting Russell's Paradox.

The absence of a universal set—indeed, the open-endedness of the hierarchy of sets—is one of the most fundamental ideas behind the cumulative-iterative conception.

Here are a few more consequences of Separation and Extensionality.

Proposition 7.3. *If any set exists, then $\emptyset$ exists.*

Proof. If A is a set, $\emptyset = \{x \in A : x \neq x\}$ exists by Separation. □

Remark 34 (Empty Set)*.* The existence of the empty set is thus *derived* from Separation (given the existence of at least one set). In many formulations of ZFC, the Empty Set axiom is listed separately; in our development it follows from AX-SET006 (Separation, §).

Remark 35 (Intersection existence)*.* Given Separation, if $A \neq \emptyset$, then $\bigcap A = \{x : (\forall y \in A)\, x \in y\}$ exists. For let $c \in A$; then $\bigcap A = \{x \in c : (\forall y \in A)\, x \in$

$y\}$, which exists by Separation. Note that $\bigcap\emptyset$ would be the universal set (vacuously), so the condition $A \neq \emptyset$ is essential. More generally, definitions of the form $C = \bigcap\{X : \varphi(X)\}$ are justified whenever some witness S with $\varphi(S)$ can be exhibited: one simply defines $C = \{x \in S : \forall X(\varphi(X) \to x \in X)\}$ using Separation.

7.2.3 Pairing

Axiom (Pairs)**.** For any sets a, b, the set $\{a, b\}$ exists.

$$\forall a \forall b \exists P \forall x(x \in P \leftrightarrow (x = a \vee x = b))$$

Here is how to justify this axiom, using the iterative conception. Suppose a is available at stage S, and b is available at stage T. Let M be whichever of stages S and T comes later. Then since a and b are both available at stage M, the set $\{a, b\}$ is a possible collection available at any stage after M.

But why assume that there *are* any stages after M? If there are none, then our justification will fail. So, to justify Pairs, we add another principle to the story:

Stages-keep-going. There is no last stage.

Even if this principle was not stated explicitly in the story of stages, it fits well with the basic idea that sets are formed in stages. We accept it in what follows, and with it, the Axiom of Pairs.

Remark 36 (Consequences of Pairing). For any sets a and b, the following sets exist:

1. $\{a\}$ (the singleton): by Pairs, $\{a, a\}$ exists, which is $\{a\}$ by Extensionality.
2. $a \cup b$ (binary union): by Pairs, $\{a, b\}$ exists; now $a \cup b = \bigcup\{a, b\}$ exists by Union (see below).
3. $\langle a, b\rangle$ (the ordered pair): $\{a\}$ exists by (1); $\{a, b\}$ by Pairs; so $\{\{a\}, \{a, b\}\} = \langle a, b\rangle$ exists by Pairs again.

7.2.4 Union

Axiom (Union)**.** For any set A, the set $\bigcup A = \{x : (\exists b \in A)\, x \in b\}$ exists.

$$\forall A \exists U \forall x(x \in U \leftrightarrow (\exists b \in A) x \in b)$$

This axiom is also justified by the cumulative-iterative conception. Let A be a set, so A is formed at some stage S (by *Stages-are-key*). Every member of A was formed *before* S (by *Stages-accumulate*); so, reasoning similarly, every member of every member of A was formed before S. Thus all of *those* sets are available before S, to be formed into a set at S. And that set is just $\bigcup A$.

7.2.5 Power Set

Axiom (Power Set)**.** For any set A, the set $\wp(A) = \{x : x \subseteq A\}$ exists.

$$\forall A \exists P \forall x (x \in P \leftrightarrow (\forall z \in x) z \in A)$$

Our justification is straightforward. Suppose A is formed at stage S. Then all of A's members were available before S (by *Stages-accumulate*). So, reasoning as in our justification for Separation, every subset of A is formed by stage S. So they are all available, to be formed into a single set, at any stage after S. And we know that there is some such stage, since S is not the last stage (by *Stages-keep-going*). So $\wp(A)$ exists.

Remark 37 (Cartesian products). Given any sets A, B, their Cartesian product $A \times B$ exists. This follows because $\wp(\wp(A \cup B))$ exists by Power Set, and $A \times B$ can be carved out by Separation: $A \times B = \{z \in \wp(\wp(A \cup B)) : (\exists x \in A)(\exists y \in B)\, z = \langle x, y \rangle\}$.

7.2.6 Infinity

We already have enough axioms to ensure that there are infinitely many sets (if there are any). For suppose some set exists, and so $\emptyset$ exists (by Proposition 7.3). Now for any set x, the set $x \cup \{x\}$ exists by Remark 36. So, applying this a few times, we obtain sets as follows:

0. $\emptyset$
1. $\{\emptyset\}$
2. $\{\emptyset, \{\emptyset\}\}$
3. $\{\emptyset, \{\emptyset\}, \{\emptyset, \{\emptyset\}\}\}$
4. $\{\emptyset, \{\emptyset\}, \{\emptyset, \{\emptyset\}\}, \{\emptyset, \{\emptyset\}, \{\emptyset, \{\emptyset\}\}\}\}$

and we can check that each of these sets is distinct. It is not hard to verify that the set labelled "n" has exactly n members, and (intuitively) is formed at the nth stage.

But this gives us *infinitely many* sets without guaranteeing that there is an *infinite set*, i.e., a set with infinitely many members. And this really matters: unless we can find a (Dedekind) infinite set, we cannot construct a Dedekind algebra to serve as the natural numbers (compare PRIM-BST012, Dedekind algebra, Chapter 1).

The axioms we have laid down so far do *not* guarantee the existence of any infinite set. So we lay down a new axiom:

Axiom (Infinity)**.** There is a set, I, such that $\emptyset \in I$ and $x \cup \{x\} \in I$ whenever $x \in I$.

$$\exists I((\exists o \in I)\forall x\, x \notin o \land \\ (\forall x \in I)(\exists s \in I)\forall z(z \in s \leftrightarrow (z \in x \lor z = x)))$$

It is easy to see that the set I given to us by the Axiom of Infinity is Dedekind infinite. Its distinguished element is $\emptyset$, and the injection on I is given by $s(x) = x \cup \{x\}$.

Definition 7.4 (ω)**.** Let I be any set given to us by the Axiom of Infinity. Let s be the function $s(x) = x \cup \{x\}$. Let $\omega = \text{clo}_s(\emptyset)$. We call the members of ω the *natural numbers*, and say that n is the result of n-many applications of s to $\emptyset$.

To justify the Axiom of Infinity, we add another principle:

> *Stages-hit-infinity*. There is an infinite stage. That is, there is a stage which (a) is not the first stage, (b) has some stages before it, but (c) has no immediate predecessor.

The Axiom of Infinity follows: natural number n is formed at stage n, so ω is formed at the first infinite stage. Unlike *Stages-keep-going*, the principle *Stages-hit-infinity* is not “forced upon us” by the iterative conception—it seems perfectly coherent to think that the stages are ordered like the natural numbers. We simply accept *Stages-hit-infinity* in what follows.

Remark 38 ($\mathbf{Z}^-$: A Milestone). The theory $\mathbf{Z}^-$ has these axioms: Extensionality, Union, Pairs, Power Set, Infinity, and all instances of the Separation scheme. The name stands for *Zermelo* set theory (minus Foundation, which we will come to below). Zermelo essentially formulated this theory in 1908. $\mathbf{Z}^-$ is powerful enough to carry out an enormous amount of mathematics; in particular, the naive set-theoretic constructions of Chapter 1 can be made rigorous within $\mathbf{Z}^-$.

7.2.7 Replacement

In order to prove that every well-ordering is isomorphic to some ordinal (a key result for ordinal theory in SET.3), we need a new axiom that goes beyond the power of $\mathbf{Z}^-$.

Axiom (Scheme of Replacement)**.** For any formula $\varphi(x, y)$, the following is an axiom:

$$\text{for any } A, \text{if } (\forall x \in A)\exists! y\, \varphi(x, y), \text{then } \{y : (\exists x \in A)\varphi(x, y)\} \text{ exists.}$$

As with Separation, this is a scheme: it yields infinitely many axioms, for each of the infinitely many different φ's. It can equally well be written thus:

For any formula $\varphi(x, y)$ which does not contain "B", the following is an axiom:

$$\forall A[(\forall x \in A)\exists! y\, \varphi(x, y) \rightarrow \exists B \forall y(y \in B \leftrightarrow (\exists x \in A)\varphi(x, y))]$$

On first encounter this is quite a tangled formula. The following quick consequence gives a *clearer* expression to the intuitive idea: for any term $\tau(x)$ and any set A, the set $\{\tau(x) : x \in A\} = \{y : (\exists x \in A)\, y = \tau(x)\}$ exists. This is because τ is a term, so $\forall x \exists! y\, \tau(x) = y$. Thus "Replacement" is a good name: given a set A, you can form a new set $\{\tau(x) : x \in A\}$ by replacing every member of A with its image under τ. Following the notation for the image of a set under a function, we might write $\tau[A]$ for $\{\tau(x) : x \in A\}$.

Crucially, τ is a *term*. It need not be a *function* in the sense of a set of ordered pairs. If f is a function (in that sense), then $f[A]$ is just a subset of $\text{ran}(f)$, already guaranteed to exist by the axioms of $\mathbf{Z}^-$. Replacement, by contrast, is a *powerful* addition to our axioms.

Remark 39 ($\mathbf{ZF}^-$: A Milestone). The theory $\mathbf{ZF}^-$ adds all instances of the Replacement scheme to $\mathbf{Z}^-$. The name stands for *Zermelo–Fraenkel* set theory (minus Foundation). Fraenkel is credited with the formulation of Replacement in 1922, although the first precise formulation was due to Skolem in the same year.

7.2.8 Foundation

We are *almost* done—but not *quite*—because nothing in $\mathbf{ZF}^-$ guarantees that *every* set is in some V_α, i.e., that every set is formed at some stage.

There is a fairly straightforward sense in which we don't *care* whether there are sets outside the hierarchy (if there are any, we can simply ignore them). But we have motivated our *concept* of set with the thought that every set is formed at some stage (see *Stages-are-key*). So we preclude the possibility of sets falling outside the hierarchy by adding a new axiom.

Since the V_αs are our stages, we might simply consider adding Regularity as an axiom:

Regularity. $\forall A \exists \alpha\, A \subseteq V_\alpha$

This would be perfectly reasonable. However, for technical reasons we instead adopt an alternative formulation:

Axiom (Foundation)**.** $(\forall A \neq \emptyset)(\exists B \in A)\, A \cap B = \emptyset$.

The connection between Foundation and Regularity requires some work. The key notion is the *transitive closure*:

Definition 7.5 (Transitive Closure)**.** For each set A, let:

$$\begin{aligned} \mathrm{cl}_0(A) &= A, \\ \mathrm{cl}_{n+1}(A) &= \bigcup \mathrm{cl}_n(A), \\ \mathrm{trcl}(A) &= \bigcup_{n<\omega} \mathrm{cl}_n(A). \end{aligned}$$

We call $\mathrm{trcl}(A)$ the *transitive closure* of A.

One can show that $A \subseteq \mathrm{trcl}(A)$ and that $\mathrm{trcl}(A)$ is a transitive set. Using Foundation and the transitive closure, one proves:

Theorem 7.6 (Foundation entails Regularity)**.** *Regularity holds in* $\mathbf{ZF}^- +$ *Foundation.*

Proof sketch. Fix A. Since $A \subseteq \mathrm{trcl}(A)$ and $\mathrm{trcl}(A)$ is transitive, it suffices to show that every transitive set is contained in some V_α. Let A be transitive, and suppose for contradiction that the set $D = \{x \in A : \forall \delta\ x \nsubseteq V_\delta\}$ is non-empty. By Foundation, there is some $B \in D$ with $D \cap B = \emptyset$. Since A is transitive, every element of B is in A but not in D, so every element of B is contained in some V_δ. Collecting these witnesses yields $B \subseteq V_\beta$ for some ordinal β, contradicting $B \in D$. $\square$

Remark 40 (Foundation–Regularity equivalence). In $\mathbf{ZF}^-$, Foundation and Regularity are equivalent. The converse direction (Regularity implies Foundation) is established using the notion of the rank of a set (see §7.5). Given $\mathbf{ZF}^-$, we can justify Foundation by noting that it is equivalent to Regularity, and Regularity follows immediately from *Stages-are-key*.

The reason we take Foundation rather than Regularity as our official axiom is that Foundation can be stated without using the V_α-hierarchy. The definition of the V_αs relies on Transfinite Recursion, whose proof employs Replacement. So while Foundation and Regularity are equivalent modulo $\mathbf{ZF}^-$, they are *not* equivalent modulo $\mathbf{Z}^-$; indeed, both $\mathbf{Z}^-$ and $\mathbf{Z}$ are too weak to define the V_αs, so Regularity (as formulated above) does not even make *sense* in $\mathbf{Z}$.

Remark 41 (**Z** and **ZF**: A Milestone). The theory **Z** adds Foundation to $\mathbf{Z}^-$. Its axioms are: Extensionality, Union, Pairs, Power Set, Infinity, Foundation, and all instances of the Separation scheme. The theory **ZF** adds Foundation to $\mathbf{ZF}^-$; equivalently, **ZF** adds all instances of Replacement to **Z**. From now on we work in **ZF** (unless otherwise stated).

Remark 42 (**ZFC**: Final Milestone). The theory **ZFC** adds Well-Ordering to **ZF**. Its axioms are: Extensionality, Union, Pairs, Power Set, Infinity, Foundation, Well-Ordering, and all instances of the Separation and Replacement schemes. The name stands for *Zermelo–Fraenkel with Choice*, because Well-Ordering turns out to be equivalent (modulo **ZF**) to the Axiom of Choice. The Well-Ordering principle (AX-SET009) is stated in SET.4 after the ordinal theory needed to define cardinals, and proven equivalent to Choice in SET.6.

7.3 Ordinals

In SET.2 we postulated that there is an infinite stage of the hierarchy (*Stages-hit-infinity*). Given *Stages-keep-going*, the stages do not stop there: at the next stage after the first infinite stage, we form all possible collections of sets available at the first infinite stage; and repeat; and repeat. Implicitly, we have invoked a notion of number that extends *beyond* the natural numbers—the notion of a *transfinite ordinal*. The aim of this section is to make that idea rigorous: we define well-orderings, introduce von Neumann's ordinals, prove their fundamental properties (including transfinite induction and the Burali-Forti paradox), and develop the machinery of transfinite recursion.

7.3.1 Well-Orderings

Definition 7.7 (Well-Ordering)**.** The relation $<$ *well-orders* A iff it meets these two conditions:

1. $<$ is connected, i.e., for all $a, b \in A$, either $a < b$ or $a = b$ or $b < a$;
2. every non-empty subset of A has a $<$-minimal element, i.e., if $\emptyset \neq X \subseteq A$ then $(\exists m \in X)(\forall z \in X)\, z \not< m$.

Proposition 7.8. *If $<$ well-orders A, then every non-empty subset of A has a unique $<$-least member, and $<$ is irreflexive, asymmetric and transitive.*

Proof. If X is a non-empty subset of A, it has a $<$-minimal element m, i.e., $(\forall z \in X)\, z \not< m$. Since $<$ is connected, $(\forall z \in X)\, m \leq z$. So m is the $<$-least element of X.

For irreflexivity, fix $a \in A$; the $<$-least element of $\{a\}$ is a, so $a \not< a$. For transitivity, if $a < b < c$, then since $\{a, b, c\}$ has a $<$-least element, $a < c$. Asymmetry follows from irreflexivity and transitivity. □

Proposition 7.9 (Well-Ordering Induction)**.** *If $<$ well-orders A, then for any formula $\varphi(x)$:*

$$\text{if } (\forall a \in A)((\forall b < a)\varphi(b) \to \varphi(a)), \text{ then } (\forall a \in A)\varphi(a).$$

Proof. Suppose $\neg(\forall a \in A)\varphi(a)$, i.e., $X = \{x \in A : \neg\varphi(x)\} \neq \emptyset$. Then X has a $<$-minimal element, a. So $(\forall b < a)\varphi(b)$ but $\neg\varphi(a)$. □

This last property should remind the reader of the principle of strong induction on the naturals: if $(\forall n \in \omega)((\forall m < n)\varphi(m) \to \varphi(n))$, then $(\forall n \in \omega)\varphi(n)$. It is this property that makes well-ordering such a *robust* notion.

7.3.2 Order-Isomorphisms

Definition 7.10 (Order-Isomorphism)**.** A *well-ordering* is a pair $\langle A, < \rangle$ such that $<$ well-orders A. The well-orderings $\langle A, < \rangle$ and $\langle B, \lessdot \rangle$ are *order-isomorphic* iff there is a bijection $f\colon A \to B$ such that $x < y$ iff $f(x) \lessdot f(y)$. In this case, we write $\langle A, < \rangle \cong \langle B, \lessdot \rangle$, and say that f is an *order-isomorphism* (or simply an *isomorphism*).

Definition 7.11 (Initial Segment)**.** When $\langle A, < \rangle$ is a well-ordering with $a \in A$, let $A_a = \{x \in A : x < a\}$. We say that A_a is a proper *initial segment* of A. Let $<_a$ be the restriction of $<$ to A_a^2.

Lemma 7.12. *If $\langle A, < \rangle$ is a well-ordering with $a \in A$, then $\langle A, < \rangle \ncong \langle A_a, <_a \rangle$.*

Proof. For reductio, suppose $f\colon A \to A_a$ is an isomorphism. Since f is a bijection and $A_a \subsetneq A$, let $b \in A$ be the $<$-least element such that $b \neq f(b)$. One shows that $(\forall x \in A)(x < b \leftrightarrow x < f(b))$, from which $b = f(b)$ by the extensionality of strict linear orders, completing the reductio. □

Lemma 7.13. *Let $\langle A, <\rangle$ and $\langle B, \lessdot\rangle$ be well-orderings. If $f\colon A \to B$ is an isomorphism and $a \in A$, then $f{\restriction}_{A_a} : A_a \to B_{f(a)}$ is an isomorphism.*

Lemma 7.14. *Let $\langle A, <\rangle$ and $\langle B, \lessdot\rangle$ be well-orderings. If $\langle A_{a_1}, <_{a_1}\rangle \cong \langle B_{b_1}, \lessdot_{b_1}\rangle$ and $\langle A_{a_2}, <_{a_2}\rangle \cong \langle B_{b_2}, \lessdot_{b_2}\rangle$, then $a_1 < a_2$ iff $b_1 \lessdot b_2$.*

Proof sketch. If $a_1 < a_2$, then $A_{a_1} \subsetneq A_{a_2}$. The isomorphism $A_{a_2} \cong B_{b_2}$ restricts (by Lemma 7.13) to an isomorphism $A_{a_1} \cong (B_{b_2})_{b'}$ for some $b' \lessdot b_2$. Since isomorphisms between well-orderings are unique (by Lemma 7.12), $b' = b_1$, giving $b_1 \lessdot b_2$. The converse is symmetric. □

Theorem 7.15 (Comparability of Well-Orderings)**.** *Given any two well-orderings, one is isomorphic to an initial segment (not necessarily proper) of the other.*

Proof sketch. Let $\langle A, <\rangle$ and $\langle B, \lessdot\rangle$ be well-orderings. Using Separation, let

$$f = \{\langle a, b\rangle \in A \times B : \langle A_a, <_a\rangle \cong \langle B_b, \lessdot_b\rangle\}.$$

By Lemma 7.14, f preserves order. One shows that $\mathrm{dom}(f)$ is an initial segment of A and $\mathrm{ran}(f)$ is an initial segment of B. If both were *proper* initial segments, say $\mathrm{dom}(f) = A_a$ and $\mathrm{ran}(f) = B_b$, then $f\colon A_a \to B_b$ would be an isomorphism, forcing $\langle a, b\rangle \in f$—a contradiction. □

7.3.3 Von Neumann's Ordinals

Theorem 7.15 gives rise to a thought: we could introduce certain objects, called *order types*, to go proxy for the well-orderings. We would hope to secure:

$$\begin{aligned}&\mathrm{ord}(A,<) = \mathrm{ord}(B,\lessdot) \text{ iff } \langle A,<\rangle \cong \langle B,\lessdot\rangle\\ &\mathrm{ord}(A,<) < \mathrm{ord}(B,\lessdot) \text{ iff } \langle A,<\rangle \cong \langle B_b,\lessdot_b\rangle \text{ for some } b \in B\end{aligned}$$

The most common way to achieve this—and the approach we follow—is to define order types via certain *canonical* well-ordered sets, first introduced by von Neumann:

Definition 7.16 (Transitive Set)**.** The set A is *transitive* iff $(\forall x \in A)\, x \subseteq A$.

Definition 7.17 (Ordinal)**.** A is an *ordinal* iff A is transitive and well-ordered by $\in$.

In what follows, we use Greek letters for ordinals. It follows immediately from the definition that if α is an ordinal, then $\langle \alpha, \in_\alpha \rangle$ is a well-ordering, where $\in_\alpha = \{\langle x, y\rangle \in \alpha^2 : x \in y\}$. So, abusing notation, we can say that α *itself* is a well-ordering.

Here are our first few ordinals:

$$\emptyset, \quad \{\emptyset\}, \quad \{\emptyset, \{\emptyset\}\}, \quad \{\emptyset, \{\emptyset\}, \{\emptyset, \{\emptyset\}\}\}, \quad \ldots$$

These are exactly the sets that appeared in our Axiom of Infinity, i.e., in the definition of ω (Definition 7.4). This is no coincidence: von Neumann's construction treats natural numbers as ordinals, but allows for transfinite ordinals too.

7.3.4 Basic Properties of the Ordinals

Lemma 7.18. *Every element of an ordinal is an ordinal.*

Proof. Let α be an ordinal with $b \in \alpha$. Since α is transitive, $b \subseteq \alpha$. So $\in$ well-orders b as $\in$ well-orders α.

To see that b is transitive, suppose $x \in c \in b$. So $c \in \alpha$ as $b \subseteq \alpha$. Again, as α is transitive, $c \subseteq \alpha$, so that $x \in \alpha$. So $x, c, b \in \alpha$. Since $\in$ well-orders α, $\in$ is transitive on α by Proposition 7.8. Hence $x \in c \in b$ gives $x \in b$. Generalising, $c \subseteq b$. □

Theorem 7.19 (Transfinite Induction)**.** *For any formula $\varphi(x)$:*

$$\text{if } \exists\alpha\varphi(\alpha), \text{ then } \exists\alpha(\varphi(\alpha) \wedge (\forall\beta \in \alpha)\neg\varphi(\beta))$$

where the displayed quantifiers are implicitly restricted to ordinals.

Proof. Suppose $\varphi(\alpha)$, for some ordinal α. If $(\forall\beta \in \alpha)\neg\varphi(\beta)$, then we are done. Otherwise, as α is an ordinal, it has some $\in$-least element which is φ, and this is an ordinal by Lemma 7.18. □

We can equally express Transfinite Induction as the scheme:

$$\text{if } \forall\alpha((\forall\beta \in \alpha)\varphi(\beta) \to \varphi(\alpha)), \text{ then } \forall\alpha\varphi(\alpha).$$

Theorem 7.20 (Trichotomy)**.** *$\alpha \in \beta \lor \alpha = \beta \lor \beta \in \alpha$, for any ordinals α and β.*

Proof. The proof is by double induction, using Transfinite Induction (Theorem 7.19) twice. Say that x is *comparable* with y iff $x \in y \lor x = y \lor y \in x$.

For induction, suppose that every ordinal in α is comparable with *every* ordinal. For further induction, suppose that α is comparable with every ordinal in β. We show that α is comparable with β. By induction on β, it follows that α is comparable with every ordinal; and by induction on α, *every* ordinal is comparable with *every* ordinal. It suffices to assume $\alpha \notin \beta$ and $\beta \notin \alpha$, and show $\alpha = \beta$.

To show $\alpha \subseteq \beta$: fix $\gamma \in \alpha$; this is an ordinal by Lemma 7.18. By the first induction hypothesis, γ is comparable with β. But if $\gamma = \beta$ or $\beta \in \gamma$, then $\beta \in \alpha$ (using transitivity of α if necessary), contrary to assumption; so $\gamma \in \beta$. Similar reasoning shows $\beta \subseteq \alpha$. So $\alpha = \beta$. □

We sometimes write $\alpha < \beta$ rather than $\alpha \in \beta$, since $\in$ behaves as an ordering relation on the ordinals.

Corollary 7.21. *A is an ordinal iff A is a transitive set of ordinals.*

Proof. Left-to-right. By Lemma 7.18. *Right-to-left.* If A is a transitive set of ordinals, then $\in$ well-orders A by Transfinite Induction (Theorem 7.19) and Trichotomy (Theorem 7.20). □

Now, we glossed Theorems 7.19 and 7.20 as telling us that $\in$ well-orders the ordinals. However, we must be cautious, thanks to the following result:

Theorem 7.22 (Burali-Forti Paradox)**.** *There is no set of all the ordinals.*

Proof. For reductio, suppose O is the set of all ordinals. If $\alpha \in \beta \in O$, then α is an ordinal by Lemma 7.18, so $\alpha \in O$. So O is transitive, and hence O is an ordinal by Corollary 7.21. Hence $O \in O$, contradicting irreflexivity (Proposition 7.8). □

The Burali-Forti paradox shows that the ordinals form a *proper class* (see Remark 33). Ordinals are sets which are individually well-ordered by membership, and collectively well-ordered by membership, without collectively constituting a set.

7.3.5 Ordinals as Order-Types

Armed with Replacement (AX-SET007, §), and so now working in $\mathbf{ZF}^-$, we can prove the key representation theorem:

Theorem 7.23 (Ordinal Representation)**.** *Every well-ordering is isomorphic to a unique ordinal.*

Proof. Let $\langle A, < \rangle$ be a well-ordering. By Theorem 7.20 and Lemma 7.12, it is isomorphic to at most one ordinal. For reductio, suppose it is not isomorphic to *any* ordinal. "Make $\langle A, < \rangle$ as small as possible": if some proper initial segment $\langle A_a, <_a \rangle$ is not isomorphic to any ordinal, let a be least with that property and set $B = A_a$; otherwise let $B = A$.

By construction, every proper initial segment of B is isomorphic to some (unique) ordinal. By Replacement, the following set exists and is a function:

$$f = \{\langle \beta, b \rangle : b \in B \text{ and } \beta \cong \langle B_b, \lessdot_b \rangle\}$$

Clearly $\text{ran}(f) = B$. By Lemma 7.14, f preserves ordering. To show $\text{dom}(f)$ is an ordinal, by Corollary 7.21 it suffices to show that $\text{dom}(f)$ is transitive: if $\beta \in \text{dom}(f)$, i.e., $\beta \cong \langle B_b, \lessdot_b \rangle$ for some b, and $\gamma \in \beta$, then $\gamma \in \text{dom}(f)$ by Lemma 7.13; so $\beta \subseteq \text{dom}(f)$. This is a contradiction. □

This licenses the following definition:

Definition 7.24 (Order Type)**.** If $\langle A, < \rangle$ is a well-ordering, then its *order type*, $\text{ord}(A, <)$, is the unique ordinal α such that $\langle A, < \rangle \cong \alpha$.

Corollary 7.25. *Where $\langle A, < \rangle$ and $\langle B, \lessdot \rangle$ are well-orderings:*

$$\begin{aligned} \text{ord}(A,<) = \text{ord}(B,\lessdot) &\textit{ iff } \langle A,<\rangle \cong \langle B,\lessdot\rangle \\ \text{ord}(A,<) \in \text{ord}(B,\lessdot) &\textit{ iff } \langle A,<\rangle \cong \langle B_b,\lessdot_b\rangle \textit{ for some } b \in B \end{aligned}$$

Proof. The first claim holds by Trichotomy and Lemma 7.12. For the second, let $\text{ord}(A, <) = \alpha$ and $\text{ord}(B, \lessdot) = \beta$, and let $f : \beta \to \langle B, \lessdot \rangle$ be an isomorphism. Then:

$$\begin{aligned} \alpha \in \beta &\text{ iff } f\restriction_\alpha : \alpha \to B_{f(\alpha)} \text{ is an isomorphism} \\ &\text{ iff } \langle A,<\rangle \cong \langle B_{f(\alpha)}, \lessdot_{f(\alpha)} \rangle \\ &\text{ iff } \langle A,<\rangle \cong \langle B_b, \lessdot_b\rangle \text{ for some } b \in B \end{aligned}$$

by Lemmas 7.13 and 7.14. □

7.3.6 Successor and Limit Ordinals

Definition 7.26 (Successor and Limit Ordinal)**.** For any ordinal α, its *successor* is $\alpha^+ = \alpha \cup \{\alpha\}$. We say that α is a *successor* ordinal if $\beta^+ = \alpha$ for some ordinal β. We say that α is a *limit* ordinal iff α is neither empty nor a successor ordinal.

Proposition 7.27. *For any ordinal α: (1) $\alpha \in \alpha^+$; (2) α^+ is an ordinal; (3) there is no ordinal β such that $\alpha \in \beta \in \alpha^+$.*

Proof. Trivially, $\alpha \in \alpha \cup \{\alpha\} = \alpha^+$. Equally, α^+ is a transitive set of ordinals, hence an ordinal by Corollary 7.21. And $\alpha \in \beta \in \alpha^+$ is impossible, since then either $\beta \in \alpha$ or $\beta = \alpha$, contradicting irreflexivity. □

Theorem 7.28 (Simple Transfinite Induction)**.** *Let $\varphi(x)$ be a formula such that: (1) $\varphi(\emptyset)$; (2) for any ordinal α, if $\varphi(\alpha)$ then $\varphi(\alpha^+)$; and (3) if α is a limit ordinal and $(\forall \beta \in \alpha)\varphi(\beta)$, then $\varphi(\alpha)$. Then $\forall \alpha\, \varphi(\alpha)$.*

Proof. Suppose there is some ordinal which is $\neg\varphi$; let γ be the least such ordinal. Then either $\gamma = \emptyset$, or $\gamma = \alpha^+$ for some α such that $\varphi(\alpha)$, or γ is a limit ordinal and $(\forall \beta \in \gamma)\varphi(\beta)$. In each case, we obtain a contradiction. □

Definition 7.29 (Least Strict Upper Bound)**.** If X is a set of ordinals, then $\text{lsub}(X) = \bigcup_{\alpha \in X} \alpha^+$.

Proposition 7.30. *If X is a set of ordinals,* $\text{lsub}(X)$ *is the least ordinal greater than every ordinal in X.*

Proof. Let $Y = \{\alpha^+ : \alpha \in X\}$, so $\text{lsub}(X) = \bigcup Y$. Since ordinals are transitive and every element of an ordinal is an ordinal, $\text{lsub}(X)$ is a transitive set of ordinals, hence an ordinal by Corollary 7.21.

If $\alpha \in X$, then $\alpha^+ \in Y$, so $\alpha^+ \subseteq \bigcup Y = \text{lsub}(X)$, hence $\alpha \in \text{lsub}(X)$. So $\text{lsub}(X)$ is strictly greater than every ordinal in X. Conversely, if $\alpha \in \text{lsub}(X)$, then $\alpha \in \beta^+ \in Y$ for some $\beta \in X$, so $\alpha \leq \beta \in X$. So $\text{lsub}(X)$ is the *least* strict upper bound on X. □

7.3.7 Transfinite Recursion

The overarching moral of this subsection is that Transfinite Induction plus Replacement guarantee the legitimacy of several versions of transfinite recursion.

Definition 7.31 (α-Approximation)**.** Let $\tau(x)$ be a term; let f be a function; let α be an ordinal. We say that f is an *α-approximation* for τ iff $\text{dom}(f) = \alpha$ and $(\forall \beta \in \alpha)\, f(\beta) = \tau(f{\restriction_\beta})$.

Lemma 7.32 (Bounded Recursion)**.** *For any term $\tau(x)$ and any ordinal α, there is a unique α-approximation for τ.*

Proof. We first establish uniqueness. Let g and h be γ- and δ-approximations, respectively. A transfinite induction on their arguments shows $g(\beta) = h(\beta)$ for any $\beta \in \text{dom}(g) \cap \text{dom}(h) = \min(\gamma, \delta)$. So approximations are unique (if they exist) and agree on all values.

To establish existence, we use Simple Transfinite Induction (Theorem 7.28) on ordinals $\delta \leq \alpha$.

The empty function is trivially an $\emptyset$-approximation.

If g is a γ-approximation, then $g \cup \{\langle \gamma, \tau(g)\rangle\}$ is a γ^+-approximation.

If γ is a limit ordinal and g_δ is a δ-approximation for all $\delta < \gamma$, let $g = \bigcup_{\delta \in \gamma} g_\delta$. This is a function since the various g_δs agree on all values. And if $\delta \in \gamma$ then $g(\delta) = g_{\delta^+}(\delta) = \tau(g_{\delta^+}{\restriction_\delta}) = \tau(g{\restriction_\delta})$. □

Theorem 7.33 (General Recursion)**.** *For any term $\tau(x)$, we can explicitly define a term $\sigma(x)$ such that $\sigma(\alpha) = \tau(\sigma{\restriction_\alpha})$ for any ordinal α.*

Proof. For each α, by Lemma 7.32 there is a unique α-approximation, f_α, for τ. Define $\sigma(\alpha)$ as $f_{\alpha^+}(\alpha)$. Then:

$$\begin{aligned}\sigma(\alpha) &= f_{\alpha^+}(\alpha)\\ &= \tau(f_{\alpha^+}{\restriction_\alpha})\\ &= \tau(\{\langle \beta, f_{\alpha^+}(\beta)\rangle : \beta \in \alpha\})\\ &= \tau(\{\langle \beta, f_{\beta^+}(\beta)\rangle : \beta \in \alpha\})\\ &= \tau(\sigma{\restriction_\alpha})\end{aligned}$$

noting that $f_{\beta^+}(\beta) = f_{\alpha^+}(\beta)$ for all $\beta < \alpha$, as in Lemma 7.32. □

Note that Theorem 7.33 is a *schema*. Crucially, we cannot expect σ to define a function (i.e., a set), since then $\text{dom}(\sigma)$ would be the set of all ordinals, contradicting Burali-Forti (Theorem 7.22).

Theorem 7.34 (Simple Recursion). *For any terms $\tau(x)$ and $\theta(x)$ and any set A, we can explicitly define a term $\sigma(x)$ such that:*

$$\begin{aligned} \sigma(\emptyset) &= A \\ \sigma(\alpha^+) &= \tau(\sigma(\alpha)) && \text{for any ordinal } \alpha \\ \sigma(\alpha) &= \theta(\text{ran}(\sigma{\restriction}_\alpha)) && \text{when } \alpha \text{ is a limit ordinal} \end{aligned}$$

Proof. Define a term $\xi(x)$ by:

$$\xi(x) = \begin{cases} A & \text{if } x \text{ is not a function whose} \\ & \quad \text{domain is an ordinal; otherwise:} \\ \tau(x(\alpha)) & \text{if } \text{dom}(x) = \alpha^+ \\ \theta(\text{ran}(x)) & \text{if } \text{dom}(x) \text{ is a limit ordinal} \end{cases}$$

By Theorem 7.33, there is a term $\sigma(x)$ such that $\sigma(\alpha) = \xi(\sigma{\restriction}_\alpha)$ for every ordinal α; moreover, $\sigma{\restriction}_\alpha$ is a function with domain α. A Simple Transfinite Induction (Theorem 7.28) confirms: $\sigma(\emptyset) = \xi(\emptyset) = A$; $\sigma(\alpha^+) = \xi(\sigma{\restriction}_{\alpha^+}) = \tau(\sigma(\alpha))$; and when α is a limit, $\sigma(\alpha) = \xi(\sigma{\restriction}_\alpha) = \theta(\text{ran}(\sigma{\restriction}_\alpha))$. □

7.3.8 Hartogs' Lemma

Lemma 7.35 (Hartogs' Lemma, in **ZF**). *For any set A, there is an ordinal α such that $\alpha \not\preceq A$.*

Proof sketch. Using Separation, consider:

$$C = \{\langle B, R\rangle : B \subseteq A \text{ and } \langle B, R\rangle \text{ is a well-ordering}\}.$$

Using Replacement and Theorem 7.23, form $\alpha = \{\text{ord}(B, R) : \langle B, R\rangle \in C\}$. By Corollary 7.21, α is an ordinal (it is a transitive set of ordinals). If there were an injection $f\colon \alpha \to A$, then $\alpha = \text{ord}(\text{ran}(f), R)$ for a suitable R, giving $\alpha \in \alpha$—a contradiction. □

Remark 43 (Hartogs' Number). The ordinal α produced in the proof is called the *Hartogs number* of A, sometimes written $\aleph(A)$. It is the least ordinal that does not inject into A.

7.4 Cardinals

In SET.3, we introduced ordinals to calibrate *well-orderings*. Two well-orderings have the same order type iff they are isomorphic. We now turn to a simpler notion: the *size* of a set. Two sets have the same size iff they are equinumerous, i.e., iff there is a bijection between them (see Chapter 1). Just as we introduced ordinals to calibrate order types, we now introduce *cardinals* to calibrate size. Writing $|X|$ for the cardinality of X, we want cardinals to satisfy *Cantor's Principle*:

$$|A| = |B| \text{ iff } A \approx B.$$

7.4.1 Cardinals as Ordinals

Our theory of cardinals makes shameless use of our theory of ordinals: we define cardinals as certain specific ordinals.

Definition 7.36 (Cardinal Number)**.** If A can be well-ordered, then $|A|$ is the least ordinal γ such that $A \approx \gamma$. For any ordinal γ, we say that γ is a *cardinal* iff $\gamma = |\gamma|$.

There is a snag with Definition 7.36: we would like $|A|$ to exist for *every* set A, but the definition begins with a conditional—"if A can be well-ordered". If some set cannot be well-ordered, the definition fails to define $|A|$. So we need a guarantee that every set can be well-ordered:

Axiom (Well-Ordering)**.** Every set can be well-ordered.

This guarantee is unavailable in **ZF** alone. The Well-Ordering axiom is the final axiom of **ZFC** (see Remark 42). Its equivalence to the Axiom of Choice will be established in SET.6 (Theorem 7.70).

Using Well-Ordering, it is straightforward to show that cardinals exist and behave well:

Lemma 7.37. *For every set A: (1) $|A|$ exists and is unique; (2) $|A| \approx A$; (3) $|A|$ is a cardinal, i.e., $|A| = ||A||$.*

Proof. Fix A. By Well-Ordering (AX-SET009), there is a well-ordering $\langle A, R\rangle$. By Theorem 7.23, $\langle A, R\rangle$ is isomorphic to a unique ordinal β, so $A \approx \beta$. By Transfinite Induction, there is a uniquely least ordinal γ such that $A \approx \gamma$. So $|A| = \gamma$, establishing (1) and (2). For (3), if $\delta \in \gamma$ then $\delta \prec A$ by choice of γ, so also $\delta \prec \gamma$ since equinumerosity is an equivalence relation. So $\gamma = |\gamma|$. □

Lemma 7.38. *For any sets A and B:*

$$A \approx B \text{ iff } |A| = |B|$$
$$A \preceq B \text{ iff } |A| \leq |B|$$
$$A \prec B \text{ iff } |A| < |B|$$

Proof. We prove left-to-right of the second claim; the other cases are similar. If $A \preceq B$, there is an injection $A \to B$. By Lemma 7.37, there are bijections $|A| \to A$ and $B \to |B|$. Composing, we obtain an injection $|A| \to |B|$, giving $|A| \leq |B|$. □

Remark 44 (Cantor's Principle). Lemma 7.38 guarantees Cantor's Principle: $|A| = |B|$ iff $A \approx B$. It also yields a quick re-proof of Schröder–Bernstein: if $A \preceq B$ and $B \preceq A$ then $|A| \leq |B|$ and $|B| \leq |A|$, so $|A| = |B|$ by Trichotomy. (This implicitly uses Replacement and Well-Ordering; the proof in Chapter 1 works in $\mathbf{Z}^-$.)

7.4.2 Finite, Infinite, and Uncountable Cardinals

Definition 7.39 (Finite and Infinite Sets)**.** We say that A is *finite* iff $|A| \in \omega$, i.e., $|A|$ is a natural number. Otherwise, A is *infinite*.

Note that this definition assumes **ZFC**, since we need Well-Ordering to guarantee $|A|$ exists. Without Well-Ordering, there can be sets that are neither finite nor Dedekind infinite (see SET.6 for the role of Choice).

Corollary 7.40. *ω is the least infinite cardinal.*

Proof. ω is a cardinal: it is Dedekind infinite, and if $\omega \approx n$ for any $n \in \omega$, then n would be Dedekind infinite, a contradiction. Now ω is the least infinite cardinal by definition, since the finite cardinals are exactly the natural numbers. □

Theorem 7.41. *There is no largest cardinal.*

Proof sketch. For any cardinal $\mathfrak{a}$, Cantor's Theorem (THM-BST001, Chapter 1) gives $\mathfrak{a} \prec \wp(\mathfrak{a})$. By Lemma 7.37, $\mathfrak{a} < |\wp(\mathfrak{a})|$. □

Proposition 7.42. *If every member of X is a cardinal, then $\bigcup X$ is a cardinal.*

Proof. It is easy to check that $\bigcup X$ is an ordinal. If $\alpha \in \bigcup X$, then $\alpha \in \mathfrak{b} \in X$ for some cardinal $\mathfrak{b}$. Since $\mathfrak{b}$ is a cardinal, $\alpha \prec \mathfrak{b}$. Since $\mathfrak{b} \subseteq \bigcup X$, we have $\mathfrak{b} \preceq \bigcup X$, and so $\alpha \not\approx \bigcup X$. Generalising, $\bigcup X$ is a cardinal. □

7.4.3 Cardinals without Well-Ordering

Remark 45 (Tarski–Scott Trick). Cardinals can be developed *without* Well-Ordering using the *Tarski–Scott trick*: for any formula $\varphi(x)$, let $[x : \varphi(x)]$ be the set of all x of least possible rank such that $\varphi(x)$, where the *rank* of a set is its position in the cumulative hierarchy $V_0 \subset V_1 \subset \cdots$ (see §7.5). Working in **ZF**, one defines the TS-cardinality of A as $\text{tsc}(A) = [x : A \approx x]$. This yields a well-behaved notion of cardinality without assuming Well-Ordering, at the cost of defining cardinals as *sets of sets* rather than ordinals.

7.5 Advanced Topics

We now develop the major advanced topics of formal set theory: the von Neumann hierarchy V_α, ordinal arithmetic, cardinal arithmetic, the Continuum Hypothesis, and the role of the Axiom of Choice.

7.5.1 The Cumulative Hierarchy

With the machinery of transfinite recursion in hand, we can give a formal, *internal* characterisation of the stages of the hierarchy.

Definition 7.43 (Von Neumann Hierarchy)**.**

$$\begin{aligned} V_\varnothing &= \varnothing \\ V_{\alpha^+} &= \wp(V_\alpha) && \text{for any ordinal } \alpha \\ V_\alpha &= \bigcup_{\gamma<\alpha} V_\gamma && \text{when } \alpha \text{ is a limit ordinal} \end{aligned}$$

This definition is legitimate by Simple Recursion (Theorem 7.34): take $A = \varnothing$, $\tau(x) = \wp(x)$, and $\theta(x) = \bigcup x$.

Lemma 7.44. *For each ordinal α: (1) V_α is transitive; (2) V_α is potent (if $x \subseteq y \in V_\alpha$ then $x \in V_\alpha$); (3) if $\gamma \in \alpha$, then $V_\gamma \in V_\alpha$ (and hence $V_\gamma \subseteq V_\alpha$).*

Proof. By simultaneous transfinite induction on α. The case $\alpha = \varnothing$ is trivial. For successor $\alpha = \beta^+$: (3) if $\gamma \in \alpha$ then $V_\gamma \subseteq V_\beta$ by hypothesis, so $V_\gamma \in$

$\wp(V_\beta) = V_\alpha$; (2) if $A \subseteq B \in V_\alpha$ then $A \subseteq V_\beta$, so $A \in V_\alpha$; (1) if $x \in A \in V_\alpha$ then $x \in V_\beta$ and $x \subseteq V_\beta$ by the induction hypothesis, so $x \in V_\alpha$. For limit α: (3) if $\gamma \in \alpha$ then $V_\gamma \in V_{\gamma^+} \subseteq V_\alpha$; (1) and (2) hold because a union of transitive (resp. potent) sets is transitive (resp. potent). □

Definition 7.45 (Rank)**.** For each set A, $\mathrm{rank}(A)$ is the least ordinal α such that $A \subseteq V_\alpha$.

Ranks exist by Foundation (AX-SET008, §) and Regularity (Theorem 7.6).

Proposition 7.46. *If* $B \in A$, *then* $\mathrm{rank}(B) \in \mathrm{rank}(A)$.

Theorem 7.47 ($\in$-Induction Scheme)**.** *For any formula φ:*

$$\forall A((\forall x \in A)\varphi(x) \to \varphi(A)) \to \forall A\, \varphi(A).$$

Proof. Suppose $\neg\forall A\, \varphi(A)$. By Transfinite Induction, there is some A of least rank with $\neg\varphi(A)$. If $x \in A$ then $\mathrm{rank}(x) \in \mathrm{rank}(A)$ by Proposition 7.46, so $\varphi(x)$. Hence $(\forall x \in A)\varphi(x)$ and $\neg\varphi(A)$. □

Proposition 7.48. $\mathrm{rank}(\alpha) = \alpha$ *for any ordinal* α.

7.5.2 Ordinal Arithmetic

We define addition, multiplication, and exponentiation on ordinals. Each can be defined synthetically (via an explicit well-ordered set and its order type) or recursively (via transfinite recursion equations). Both approaches yield the same result by Theorem 7.23.

Definition 7.49 (Ordinal Addition)**.** The *disjoint sum* of A and B is $A \sqcup B = (A \times \{0\}) \cup (B \times \{1\})$. Define the *reverse lexicographic ordering* $\lhd$ on $\alpha \sqcup \beta$ by: $\langle \alpha_1, \alpha_2 \rangle \lhd \langle \beta_1, \beta_2 \rangle$ iff either $\alpha_2 \in \beta_2$, or both $\alpha_2 = \beta_2$ and $\alpha_1 \in \beta_1$. Then $\alpha + \beta = \mathrm{ord}(\alpha \sqcup \beta, \lhd)$.

Ordinal addition satisfies the following recursion equations:

$$\begin{aligned} \alpha + 0 &= \alpha \\ \alpha + (\beta + 1) &= (\alpha + \beta) + 1 \\ \alpha + \beta &= \operatorname*{lsub}_{\delta < \beta}(\alpha + \delta) && \text{if } \beta \text{ is a limit ordinal} \end{aligned}$$

Addition is associative: $\alpha + (\beta + \gamma) = (\alpha + \beta) + \gamma$. It is *not* commutative: $1 + \omega = \omega < \omega + 1$. Intuitively, placing one element *before* an ω-sequence does not change the order type (by a Hilbert's Hotel argument), but placing one element *after* it does.

Definition 7.50 (Ordinal Multiplication)**.** $\alpha \cdot \beta = \mathrm{ord}(\alpha \times \beta, \lhd)$. Equivalently, by transfinite recursion:

$$\begin{aligned} \alpha \cdot 0 &= 0 \\ \alpha \cdot (\beta + 1) &= (\alpha \cdot \beta) + \alpha \\ \alpha \cdot \beta &= \underset{\delta<\beta}{\mathrm{lsub}}(\alpha \cdot \delta) && \text{when } \beta \text{ is a limit ordinal} \end{aligned}$$

Multiplication is associative but *not* commutative: $2 \cdot \omega = \omega < \omega \cdot 2$. It distributes over addition from the right: $\alpha \cdot (\beta + \gamma) = (\alpha \cdot \beta) + (\alpha \cdot \gamma)$.

Definition 7.51 (Ordinal Exponentiation)**.** By transfinite recursion:

$$\begin{aligned} \alpha^{(0)} &= 1 \\ \alpha^{(\beta+1)} &= \alpha^{(\beta)} \cdot \alpha \\ \alpha^{(\beta)} &= \bigcup_{\delta<\beta} \alpha^{(\delta)} && \text{when } \beta \text{ is a limit ordinal} \end{aligned}$$

Ordinal exponentiation does not commute: $2^{(\omega)} = \bigcup_{n<\omega} 2^{(n)} = \omega$, whereas $\omega^{(2)} = \omega \cdot \omega$.

Lemma 7.52 (Characterisation of Infinite Ordinals)**.** *For any ordinal α, the following are equivalent: (1) $\alpha \notin \omega$; (2) $\omega \leq \alpha$; (3) $1 + \alpha = \alpha$; (4) α and $\alpha + 1$ are equinumerous; (5) α is Dedekind infinite.*

Proof. (1) $\Rightarrow$ (2): By Trichotomy. (2) $\Rightarrow$ (3): Write $\alpha = \beta + \gamma$ where β is a limit ordinal and γ is least. Then $1 + \alpha = (1 + \beta) + \gamma = \beta + \gamma = \alpha$, using the fact that $1 + \beta = \mathrm{lsub}_{\delta<\beta}(1 + \delta) = \beta$. (3) $\Rightarrow$ (4): There is a bijection $\alpha \sqcup 1 \to 1 \sqcup \alpha$; compose with the isomorphism $1 \sqcup \alpha \to \alpha$. (4) $\Rightarrow$ (5): A bijection $\alpha \sqcup 1 \to \alpha$ restricts to an injection $\alpha \to \alpha$ that is not surjective. (5) $\Rightarrow$ (1): No natural number is Dedekind infinite. $\square$

7.5.3 Cardinal Arithmetic

Since we do not need to keep track of order, cardinal arithmetic is rather easier to define than ordinal arithmetic.

Definition 7.53 (Cardinal Operations)**.** When $\mathfrak{a}$ and $\mathfrak{b}$ are cardinals:

$$\mathfrak{a} \oplus \mathfrak{b} = |\mathfrak{a} \sqcup \mathfrak{b}|$$
$$\mathfrak{a} \otimes \mathfrak{b} = |\mathfrak{a} \times \mathfrak{b}|$$
$$\mathfrak{a}^{\mathfrak{b}} = \left|{}^{\mathfrak{b}}\mathfrak{a}\right|$$

where ${}^{X}Y = \{f : f \text{ is a function } X \to Y\}$.

Cardinal addition and multiplication are commutative and associative (unlike their ordinal counterparts).

Lemma 7.54. *$|\wp(A)| = 2^{|A|}$, for any A.*

Proof. For each $B \subseteq A$, define the characteristic function $\chi_B \in {}^{A}2$ by $\chi_B(x) = 1$ if $x \in B$, and $\chi_B(x) = 0$ otherwise. The map $B \mapsto \chi_B$ is a bijection $\wp(A) \to {}^{A}2$. □

Corollary 7.55 (Cantor's Theorem in Cardinal Arithmetic)**.** *$\mathfrak{a} < 2^{\mathfrak{a}}$ for any cardinal $\mathfrak{a}$.*

Proof. From Cantor's Theorem (THM-BST001, Chapter 1) and Lemma 7.54. □

Theorem 7.56. *$|\mathbb{R}| = 2^{\omega}$.*

Proof skeleton. Show $\wp(\omega) \preceq \mathbb{R}$ and $\mathbb{R} \preceq \wp(\omega)$; apply Schröder–Bernstein to get $\mathbb{R} \approx \wp(\omega)$; then $|\mathbb{R}| = |\wp(\omega)| = 2^{\omega}$ by Lemma 7.54. □

Theorem 7.57 (Simplification of Addition and Multiplication)**.** *If $\mathfrak{a}, \mathfrak{b}$ are infinite cardinals, then $\mathfrak{a} \otimes \mathfrak{b} = \mathfrak{a} \oplus \mathfrak{b} = \max(\mathfrak{a}, \mathfrak{b})$.*

Proof sketch. The key step is showing $\alpha \approx \alpha \times \alpha$ for any infinite ordinal α. This uses the *canonical ordering* $\lhd$ on pairs: $\langle \alpha_1, \alpha_2 \rangle \lhd \langle \beta_1, \beta_2 \rangle$ iff either $\max(\alpha_1, \alpha_2) < \max(\beta_1, \beta_2)$, or they have the same max and $\alpha_1 < \beta_1$, or same max and first coordinate and $\alpha_2 < \beta_2$. This is a well-ordering of $\alpha \times \alpha$. For the least infinite ordinal α for which $\alpha \approx \alpha \times \alpha$ fails, one shows α must be a cardinal, and each segment $\mathrm{Seg}(\gamma_1, \gamma_2)$ in the canonical ordering has cardinality $< \alpha$, giving $\mathrm{ord}(\alpha \times \alpha, \lhd) \leq \alpha$, hence $\alpha \times \alpha \preceq \alpha$. The reverse injection is obvious, so Schröder–Bernstein applies.

Given this, without loss of generality let $\mathfrak{a} = \max(\mathfrak{a}, \mathfrak{b})$. Then $\mathfrak{a} \otimes \mathfrak{a} = \mathfrak{a} \leq \mathfrak{a} \oplus \mathfrak{b} \leq \mathfrak{a} \oplus \mathfrak{a} \leq \mathfrak{a} \otimes \mathfrak{a}$. □

Proposition 7.58. *Let $\mathfrak{a}$ be an infinite cardinal. For each $\beta \in \mathfrak{a}$, let X_β be a set with $|X_\beta| \leq \mathfrak{a}$. Then $\left|\bigcup_{\beta \in \mathfrak{a}} X_\beta\right| \leq \mathfrak{a}$.*

Proof sketch. Each X_β injects into $\mathfrak{a}$, so $\bigcup_\beta X_\beta$ injects into $\mathfrak{a} \times \mathfrak{a}$; by the preceding theorem, $|\mathfrak{a} \times \mathfrak{a}| = \mathfrak{a}$. □

7.5.4 Aleph and Beth Numbers

Definition 7.59 (Aleph and Beth Numbers)**.** Where $\mathfrak{a}^\oplus$ is the least cardinal strictly greater than $\mathfrak{a}$, we define two sequences by transfinite recursion:

$$\begin{aligned} \aleph_0 &= \omega & \beth_0 &= \omega \\ \aleph_{\alpha+1} &= (\aleph_\alpha)^\oplus & \beth_{\alpha+1} &= 2^{\beth_\alpha} \\ \aleph_\alpha &= \bigcup_{\beta<\alpha} \aleph_\beta & \beth_\alpha &= \bigcup_{\beta<\alpha} \beth_\beta & &\text{when } \alpha \text{ is a limit ordinal} \end{aligned}$$

The definition of $\mathfrak{a}^\oplus$ is in order: for each cardinal $\mathfrak{a}$, there is some cardinal greater than $\mathfrak{a}$ (Theorem 7.41), and Transfinite Induction gives the *least* such. The "$\aleph$" notation is due to Cantor; "$\beth$" is due to Peirce.

Proposition 7.60. *$\aleph_\alpha$ and $\beth_\alpha$ are cardinals for every ordinal α. Moreover, every infinite cardinal is an $\aleph$: if $\mathfrak{a}$ is an infinite cardinal, then $\mathfrak{a} = \aleph_\gamma$ for some unique γ.*

Proof. By transfinite induction. $\aleph_0 = \beth_0 = \omega$ is a cardinal by Corollary 7.40; successors are cardinals by definition; limits are cardinals by Proposition 7.42.

For the second claim, induct on cardinals. If $\mathfrak{a}$ is the successor of some $\mathfrak{b} = \aleph_\gamma$, then $\mathfrak{a} = \aleph_{\gamma+1}$. If $\mathfrak{a}$ is not a cardinal successor, then $\mathfrak{a} = \bigcup_{\mathfrak{b}<\mathfrak{a}} \mathfrak{b} = \bigcup_{\mathfrak{b}<\mathfrak{a}} \aleph_{\gamma_\mathfrak{b}} = \aleph_\gamma$ for a suitable limit γ. □

7.5.5 The Continuum Hypothesis

Since every infinite cardinal is an $\aleph$, we ask: is every infinite cardinal a $\beth$? If so, infinite cardinals would "play straightforwardly" with powersets:

[Generalised Continuum Hypothesis] GCH: $\aleph_\alpha = \beth_\alpha$, for all α.

If GCH held, cardinal exponentiation could be completely determined: for $\mathfrak{b} < \mathfrak{a}$, the value $\mathfrak{a}^\mathfrak{b}$ would be trapped by $\mathfrak{a} \leq \mathfrak{a}^\mathfrak{b} \leq \mathfrak{a}^\oplus$.

But GCH is a *hypothesis*, not a theorem. Gödel (1938) proved that if **ZFC** is consistent, then so is **ZFC** + GCH. The simplest non-trivial instance is:

[Continuum Hypothesis] CH: $\aleph_1 = \beth_1$.

Cohen (1963) proved that if **ZFC** is consistent, then so is **ZFC** + ¬CH. So the Continuum Hypothesis is *independent* from **ZFC**.

The Continuum Hypothesis is so-called because "the continuum" is another name for the real line $\mathbb{R}$. Theorem 7.56 tells us $|\mathbb{R}| = \beth_1$. So CH states there is no cardinal between $\aleph_0 = \beth_0$ (the cardinality of the natural numbers) and $\beth_1$ (the cardinality of the continuum).

Two observations are worth emphasising. First, it does not immediately follow from independence that CH is indeterminate in truth value: perhaps additional natural axioms will settle it. Gödel himself suggested this response. Second, the independence of CH is striking but not incredible: for all **ZFC** tells us, moving from a cardinal to its successor may involve a more refined tool than simply taking powersets.

7.5.6 The Strength of Replacement

We briefly note the strength of the Replacement scheme.

Theorem 7.61. **Z** *is consistent with the non-existence of* $\omega + \omega$.

Proof sketch. Working in **ZF**, consider $V_{\omega+\omega}$. One can show that for every axiom φ of **Z**, we have **ZF** $\vdash \varphi^{V_{\omega+\omega}}$ (where φ^M restricts all quantifiers to M). But $\omega + \omega \notin V_{\omega+\omega}$ (since $\text{rank}(\omega+\omega) = \omega+\omega$ by Proposition 7.48). So **Z** does not prove the existence of $\omega + \omega$. □

This is why Theorem 7.23 cannot be proved without Replacement: within **Z** one can define a well-ordering of order type $\omega + \omega$, but if $\omega + \omega$ does not exist, this well-ordering is not isomorphic to any ordinal.

More broadly, Replacement forces the hierarchy to be very tall.

Theorem 7.62 (Reflection Schema)**.** *For any formula* φ*:*

$$\forall\alpha\exists\beta > \alpha\, (\forall x_1, \dots, x_n \in V_\beta)(\varphi(x_1, \dots, x_n) \leftrightarrow \varphi^{V_\beta}(x_1, \dots, x_n))$$

Montague (1961) and Lévy (1960) showed that Replacement and Reflection are equivalent modulo **Z**, so adding either gives **ZF**.

Theorem 7.63. **ZF** *is not finitely axiomatizable: if $\mathcal{T}$ is finite and $\mathcal{T} \vdash$* **ZF***, then $\mathcal{T}$ is inconsistent.*

Proof sketch. By Reflection, for any finite set of sentences $\mathcal{T} \subseteq$ **ZF** there is a β such that $V_\beta \models \mathcal{T}$. If $\mathcal{T}$ axiomatised all of **ZF**, then **ZF** would prove "there exists a set model of $\mathcal{T}$," hence **ZF** $\vdash \text{Con}(\mathcal{T}) = \text{Con}(\mathbf{ZF})$. By the Second Incompleteness Theorem, **ZF** is then inconsistent. □

7.5.7 Fixed Points

The following results illustrate just how tall Replacement forces the hierarchy to be.

Proposition 7.64 ($\aleph$-Fixed Point)**.** *There is a cardinal κ such that $\kappa = \aleph_\kappa$.*

Proof. Define by recursion: $\kappa_0 = 0$; $\kappa_{n+1} = \aleph_{\kappa_n}$; $\kappa = \bigcup_{n<\omega} \kappa_n$. Then κ is a cardinal by Proposition 7.42, and $\kappa = \bigcup_{n<\omega} \kappa_{n+1} = \bigcup_{n<\omega} \aleph_{\kappa_n} = \bigcup_{\alpha<\kappa} \aleph_\alpha = \aleph_\kappa$. □

Proposition 7.65 ($\beth$-Fixed Point)**.** *There is a κ such that $\kappa = \beth_\kappa$.*

Proof. As in Proposition 7.64, using "$\beth$" in place of "$\aleph$". □

Proposition 7.66. $|V_{\omega+\alpha}| = \beth_\alpha$*. If $\omega \cdot \omega \leq \alpha$, then $|V_\alpha| = \beth_\alpha$.*

Proof sketch. By transfinite induction on α. For the base, V_ω is countable, so $|V_\omega| = \aleph_0 = \beth_0$. At successor stages, $|V_{\omega+\alpha+1}| = 2^{|V_{\omega+\alpha}|} = 2^{\beth_\alpha} = \beth_{\alpha+1}$. At limits, take the union. □

Corollary 7.67. *There is a κ such that $|V_\kappa| = \kappa$.*

Proof. Let κ be a $\beth$-fixed point (Proposition 7.65). Then $|V_\kappa| = \beth_\kappa = \kappa$ by Proposition 7.66. □

Intuitively, V_κ is "as wide as it is tall": there are as many stages beneath it as there are elements of it. This is a Tristram-Shandy phenomenon: we move from one stage to the next by taking powersets, making the hierarchy much wider with each step, yet "in the end" the width catches up with the height.

7.5.8 The Axiom of Choice

The Axiom of Well-Ordering (AX-SET009, §) has been essential to our development of cardinal arithmetic. We now discuss its justification and its relationship to other principles.

Definition 7.68 (Choice Function)**.** A function f is a *choice function* iff $f(x) \in x$ for all $x \in \text{dom}(f)$. We say f is a choice function *for* A iff $\text{dom}(f) = A \setminus \{\emptyset\}$.

Axiom (Choice)**.** Every set has a choice function.

The equivalence of Choice and Well-Ordering is established in SET.6 (Theorem 7.70).

Definition 7.69 (Zorn's Lemma)**.** A *chain* in a partially ordered set P is a totally ordered subset of P. *Zorn's Lemma* states: if every chain in a partially ordered set P has an upper bound in P, then P has a maximal element.

Zorn's Lemma is equivalent (in **ZF**) to both Choice and Well-Ordering (Theorem 7.70).

Remark 46 (Justification of Choice). One intrinsic justification appeals to the cumulative-iterative conception. Let A's elements be disjoint and non-empty. By *Stages-are-key*, A is formed at some stage S. All elements of $\bigcup A$ are available before S. By *Stages-accumulate*, every possible collection of earlier-available sets exists at S. It is certainly *possible* to select one element from each member of A—a choice set for A. So such a choice set exists. (For a careful development of this argument, see Potter (§14.8).)

Remark 47 (Countable Choice). *Countable Choice* states that every countable set has a choice function. This special case was used frequently by 19th-century mathematicians (including Dedekind and Cantor) without awareness of its role. Two notable consequences requiring Countable Choice: (1) every set is either finite or contains a countably infinite subset (Dedekind); (2) a countable union of countable sets is countable (Cantor). Both fail in **ZF** alone: Cohen proved it is consistent with **ZF** that some sets are incomparable with ω; Feferman and Levy proved it is consistent with **ZF** that a countable union of countable sets has cardinality $\beth_1$.

7.6 Theorems

We close this chapter by establishing the fundamental equivalences that unify the theory.

7.6.1 Well-Ordering iff Choice

Theorem 7.70 (in **ZF**). *The following are equivalent:*

1. *Well-Ordering (AX-SET009): every set can be well-ordered.*
2. *Choice: every set has a choice function.*
3. *Zorn's Lemma (DEF-SET010): if every chain in a partially ordered set P has an upper bound in P, then P has a maximal element.*

Proof. Well-Ordering ⇒ Choice. Let A be a set of sets. Then $\bigcup A$ exists by Union, and by Well-Ordering there is some $<$ which well-orders $\bigcup A$. Define $f(x) =$ the $<$-least member of x. This is a choice function for A.

Choice ⇒ Well-Ordering. Fix A. By Choice, there is a choice function f for $\wp(A) \setminus \{\emptyset\}$. Using Transfinite Recursion (Theorem 7.33), define:

$$g(0) = f(A)$$
$$g(\alpha) = \begin{cases} \text{stop} & \text{if } A = g[\alpha] \\ f(A \setminus g[\alpha]) & \text{otherwise} \end{cases}$$

Since f is a choice function, for each α (when defined) we have $g(\alpha) = f(A \setminus g[\alpha]) \in A \setminus g[\alpha]$, so $g(\alpha) \notin g[\alpha]$. If $g(\alpha) = g(\beta)$ then $g(\beta) \notin g[\alpha]$, i.e., $\beta \notin \alpha$, and similarly $\alpha \notin \beta$. So $\alpha = \beta$ by Trichotomy, hence g is injective.

We must stop: i.e., there is some least α with $A = g[\alpha]$. For if not, then as g is injective we would have $\alpha \prec \wp(A) \setminus \{\emptyset\}$ for every ordinal α, contradicting Hartogs' Lemma (Lemma 7.35).

Assembling these facts, g is a bijection from some ordinal to A, and can be used to well-order A.

Choice ⇔ Zorn's Lemma. This equivalence is a standard result; the proof that Choice implies Zorn's Lemma uses transfinite recursion to build a maximal chain. The converse constructs a choice function by applying Zorn's Lemma to a suitable partial order of partial choice functions. □

7.6.2 Comparability of Sets

Theorem 7.71 (in **ZF**). *The following are equivalent:*

1. *Well-Ordering.*
2. *Either $A \preceq B$ or $B \preceq A$, for any sets A and B.*

Proof. (1) ⇒ (2). Fix A and B. By Well-Ordering, there are well-orderings $\langle A, R\rangle$ and $\langle B, S\rangle$. Let $f\colon \alpha \to \langle A, R\rangle$ and $g\colon \beta \to \langle B, S\rangle$ be isomorphisms. By

Trichotomy, either $\alpha \subseteq \beta$ or $\beta \subseteq \alpha$. If $\alpha \subseteq \beta$, then $g^{-1} \circ (f\restriction_{\alpha})$ is an injection $A \to B$, so $A \preceq B$; similarly in the other case.

(2) ⇒ (1). Fix A; by Hartogs' Lemma (Lemma 7.35) there is an ordinal β with $\beta \not\preceq A$. By (2), $A \preceq \beta$. An injection $f \colon A \to \beta$ induces a well-ordering on A via $a_1 < a_2$ iff $f(a_1) \in f(a_2)$. □

As an immediate consequence: if Well-Ordering fails, then some sets are *literally incomparable* with regard to their size, and transfinite cardinal arithmetic becomes significantly more complex.

Remark 48 (Summary of SET.3–SET.6). Working in **ZF**, we developed the ordinals (von Neumann's construction) and proved the key structural theorems: Transfinite Induction (Theorem 7.19), Trichotomy (Theorem 7.20), Burali-Forti (Theorem 7.22), and Ordinal Representation (Theorem 7.23). The machinery of transfinite recursion (Theorems 7.33 and 7.34) enabled the definition of the cumulative hierarchy V_α (Definition 7.43) and ordinal arithmetic. Adding the Well-Ordering axiom (AX-SET009, §) yielded **ZFC** and a robust theory of cardinals. Cardinal arithmetic (Theorem 7.57) simplifies dramatically for infinite cardinals; but the Continuum Hypothesis (section 7.5.5) remains independent of **ZFC**. The equivalence of Well-Ordering, Choice, and Zorn's Lemma (Theorem 7.70) unifies the theory.

Chapter 8

Extensions

The preceding seven chapters establish a self-contained development of classical first-order logic: from set-theoretic foundations (CH-BST) through syntax, semantics, deduction, computation, metatheory, and formal set theory. This final chapter catalogues the principal directions in which the core framework can be extended. Each topic area is the subject of extensive treatment in the OpenLogic Project source material; we provide brief orientations and pointers.

8.1 Modal Logic

Modal logic extends propositional and first-order logic with operators expressing necessity ($\Box$) and possibility ($\Diamond$). The semantics is given by Kripke structures (relational models): a set of worlds, an accessibility relation, and a valuation at each world. A formula $\Box\varphi$ holds at a world w iff φ holds at every world accessible from w. By varying the properties of the accessibility relation—reflexive, transitive, symmetric, Euclidean—one obtains the standard normal modal systems K, T, S4, and S5. Completeness, correspondence theory (frame conditions vs. axiom schemas), and filtration-based decidability proofs constitute the core metatheory. Applied extensions include epistemic logic (knowledge and belief), temporal logic (operators for future and past), and deontic logic (obligation and permission).

Source material: OpenLogic `content/normal-modal-logic/` (69 sections) and `content/applied-modal-logic/` (16 sections).

8.2 Intuitionistic Logic

Intuitionistic logic rejects the law of excluded middle $\varphi \lor \neg\varphi$ and the double-negation elimination rule $\neg\neg\varphi \to \varphi$, restricting provability to constructive reasoning. Its semantics can be given via Kripke models (with a partial order of information states), Heyting algebras, or the Brouwer–Heyting–Kolmogorov interpretation (proofs as constructions). Natural deduction and sequent calculus formulations are obtained by restricting the classical rules (e.g., allowing at most one formula in the succedent of sequent calculus sequents, or restricting *reductio ad absurdum* in ND). The propositions-as-types correspondence (Curry–Howard isomorphism) connects intuitionistic proofs to typed lambda terms, establishing deep links between logic and computation.

Source material: OpenLogic `content/intuitionistic-logic/` (34 sections).

8.3 Many-Valued Logic

Many-valued logics generalize classical logic by allowing truth values beyond $\{\mathbb{T}, \mathbb{F}\}$. Three-valued logics (Łukasiewicz, Kleene, Priest) introduce a third value for "unknown," "undefined," or "both true and false." Continuous-valued logics (Łukasiewicz infinite-valued, Gödel logic, product logic) use the real interval $[0, 1]$ as the truth-value set. The algebraic semantics is given by MV-algebras, Heyting algebras, or BL-algebras. Key metatheoretic results include completeness theorems for various axiomatic systems and the McNaughton theorem characterizing Łukasiewicz logic functions as piecewise-linear functions on $[0, 1]$.

Source material: OpenLogic `content/many-valued-logic/` (25 sections).

8.4 Second-Order Logic

Second-order logic extends first-order logic by allowing quantification over predicate and function variables in addition to individual variables. Under the standard (or "full") semantics, second-order quantifiers range over all subsets (or all functions) on the domain. This dramatically increases expressive power: the natural numbers, the real numbers, and well-orderings are categorically axiomatizable in second-order logic. However, this power comes at a metatheoretic cost: the Completeness Theorem fails for the standard semantics (second-order validity is not axiomatizable), and the Löwenheim–Skolem theorem fails. Under Henkin semantics (where second-order quantifiers range over a specified subcollection), completeness and compactness are restored, but categorical axiomatizability is lost.

Source material: OpenLogic `content/second-order-logic/` (20 sections).

8.5 Lambda Calculus

The lambda calculus, introduced by Church in the 1930s, provides a formal system for defining and applying functions. The untyped lambda calculus uses three term constructors—variables, abstraction ($\lambda x. M$), and application ($M\,N$)—with β-reduction ($(\lambda x. M)\,N \to M[N/x]$) as the fundamental computation rule. The Church–Rosser theorem guarantees confluence: if a term reduces in two different ways, both reduction paths can be completed to a common reduct. The simply-typed lambda calculus adds type annotations ($\lambda x{:}\alpha. M$) and guarantees strong normalization (every reduction sequence terminates), establishing a correspondence with intuitionistic propositional logic via the Curry–Howard isomorphism. Extensions to System F (polymorphism), dependent types, and the calculus of constructions connect to modern proof assistants and programming language theory.

Source material: OpenLogic `content/lambda-calculus/` (44 sections).

8.6 Other Extensions

Several additional topics in the OpenLogic Project fall outside the core development but are of independent interest:

- **Counterfactual conditionals** (13 sections): Stalnaker and Lewis semantics for counterfactuals using sphere models and minimal-change semantics, addressing the failure of antecedent strengthening, contraposition, and transitivity for the counterfactual conditional.
- **Methods** (19 sections): Proof techniques and problem sets designed for introductory logic courses, including induction templates, proof strategies, and worked examples.
- **History of logic** (20 sections): Biographies of key figures (Cantor, Gödel, Turing, Tarski, Church, Gentzen, Robinson, Noether, Peter, Russell, Zermelo) and historical developments in set theory (Cantor on the line and the plane, space-filling curves, infinitesimals, pathological sets).
- **Reference material** (3 sections): Notation summaries and symbol tables.

Source material: OpenLogic `content/counterfactuals/`, `content/methods/`, `content/history/`, and `content/reference/`.

Notation Index

The following table collects the principal symbols used throughout this text. Symbols are grouped by domain; page numbers refer to the first defining occurrence.

Standard Sets

$\mathbb{N}$	Natural numbers	CH-BST
$\mathbb{Z}$	Integers	CH-BST
$\mathbb{Z}^+$	Positive integers	CH-BST
$\mathbb{Q}$	Rational numbers	CH-BST
$\mathbb{R}$	Real numbers	CH-BST
$\mathbb{B}$	Binary / Boolean values	CH-BST

Set-Theoretic Notation

$\{x : \varphi(x)\}$	Set comprehension $\{x : \varphi(x)\}$	CH-BST
$\wp(A)$	Power set of A	CH-BST
$\langle a, b\rangle$	Ordered pair / tuple	CH-BST
$A \times B$	Cartesian product	CH-BST
$\mathrm{dom}(f)$	Domain of f	CH-BST
$\mathrm{ran}(f)$	Range of f	CH-BST
$g \circ f$	Composition $g \circ f$	CH-BST
$\lvert A\rvert$	Cardinality of A	CH-BST
$A \preceq B$	A is no larger than B	CH-BST
$A \prec B$	A is strictly smaller than B	CH-BST
$A \approx B$	A and B are equinumerous	CH-BST
Id_A	Identity relation on A	CH-BST

Logical Connectives and Quantifiers

$\neg\varphi$	Negation	CH-SYN
$\varphi \wedge \psi$	Conjunction	CH-SYN
$\varphi \vee \psi$	Disjunction	CH-SYN
$\varphi \rightarrow \psi$	Material conditional	CH-SYN
$\varphi \leftrightarrow \psi$	Biconditional	CH-SYN
$\top$	Logical truth	CH-SYN
$\bot$	Logical falsum	CH-SYN
$\forall x\, \varphi(x)$	Universal quantification	CH-SYN
$\exists x\, \varphi(x)$	Existential quantification	CH-SYN
$t_1 = t_2$	Identity / equality	CH-SYN

Syntax

$\mathcal{L}$	Formal language	CH-SYN
Var	Set of variables	CH-SYN
At_0	Set of propositional variables	CH-SYN
$\mathrm{Trm}(\mathcal{L})$	Set of terms of $\mathcal{L}$	CH-SYN
$\mathrm{Frm}(\mathcal{L})$	Set of formulas of $\mathcal{L}$	CH-SYN
$\mathrm{Sent}(\mathcal{L})$	Set of sentences of $\mathcal{L}$	CH-SYN
$P(t_1, \dots, t_n)$	Atomic formula	CH-SYN
$\mathrm{SFrm}(\varphi)$	Set of subformulas of φ	CH-SYN
$\mathrm{FV}(\varphi)$	Free variables of φ	CH-SYN
$\varphi[t/x]$	Substitution of t for x in φ	CH-SYN
$\varphi \equiv \psi$	Syntactic identity	CH-SYN
$\overline{n}$	Standard numeral for n	CH-SYN

Semantics

$\mathfrak{M}$	First-order structure	CH-SEM
$\lvert\mathfrak{M}\rvert$	Domain (universe) of $\mathfrak{M}$	CH-SEM
$f^{\mathfrak{M}}$	Interpretation of f in $\mathfrak{M}$	CH-SEM
$\mathrm{Val}_s^{\mathfrak{M}}(t)$	Value of term t under assignment s	CH-SEM
$\mathfrak{M}, s \vDash \varphi$	$\mathfrak{M}$ satisfies φ under s	CH-SEM
$[\vDash /]M\varphi$	$\mathfrak{M}$ does not satisfy φ	CH-SEM
$\vDash$	Semantic consequence (entailment)	CH-SEM
$\vDash_/$	Does not entail	CH-SEM
$\mathrm{Th}(\mathfrak{M})$	Theory of $\mathfrak{M}$	CH-SEM
$\mathrm{Mod}_\Gamma()$	Class of models of Γ	CH-SEM
$\mathfrak{M} \equiv \mathfrak{N}$	Elementary equivalence	CH-SEM
$\mathfrak{M} \simeq \mathfrak{N}$	Isomorphism	CH-SEM
$\mathfrak{v}$	Propositional valuation	CH-SEM
$\mathfrak{v} \vDash \varphi$	Propositional satisfaction	CH-SEM
$\mathrm{qr}(\varphi)$	Quantifier rank of φ	CH-SEM

Deduction

$\Gamma \vdash \varphi$	φ is derivable from Γ	CH-DED
$\Gamma \vdash_/ \varphi$	φ is not derivable from Γ	CH-DED
$\vdash \varphi$	φ is a theorem	CH-DED
MP	Modus ponens	CH-DED
$\to$Intro, $\to$Elim	ND introduction / elimination rule	CH-DED
$[\varphi]^n$	Discharged assumption (label n)	CH-DED
$\Gamma \Rightarrow \Delta$	Sequent	CH-DED
$\to$L, $\to$R	SC left / right rule	CH-DED
W, C, X, Cut	Structural rules	CH-DED
$\mathbb{T}\varphi$, $\mathbb{F}\varphi$	Signed formula (tableau)	CH-DED

Theories and Formal Arithmetic

T	Named theory	CH-DED
Q	Robinson arithmetic	CH-DED
PA	Peano arithmetic	CH-DED
$=$	Definitional equality	CH-DED
$(\forall x < t)\ \varphi$	Bounded universal quantifier	CH-META
$(\exists x < t)\ \varphi$	Bounded existential quantifier	CH-META

Computability

zero, succ, add, mult	Basic recursive functions	CH-CMP
P^n_i	Projection function	CH-CMP
χ_R	Characteristic function of R	CH-CMP
$\mu x\ R(x)$	Unbounded minimization	CH-CMP
φ_e	e-th partial computable function	CH-CMP
$f(x)\downarrow$	$f(x)$ is defined (converges)	CH-CMP
$f(x)\uparrow$	$f(x)$ is undefined (diverges)	CH-CMP
$f\colon A \nrightarrow B$	Partial function from A to B	CH-CMP
$\frown$	String / sequence concatenation	CH-CMP

Gödel Numbering and Provability

$\ulcorner\varphi\urcorner$	Gödel number of φ	CH-META
$^{\#}\varphi^{\#}$	Gödel number (corner-quote style)	CH-META
$\mathsf{Prov}_{\mathbf{T}}(x)$	Provability predicate for **T**	CH-META
$\mathsf{RProv}_{\mathbf{T}}(x)$	Rosser provability predicate	CH-META
$\mathsf{Con}_{\mathbf{T}}$	Consistency statement for **T**	CH-META
$\mathsf{Prf}_{\mathbf{T}}(x,y)$	Proof predicate for **T**	CH-META

Formal Set Theory

ZF, **ZFC**	Zermelo–Fraenkel (with Choice)	CH-SET
$\mathbf{ZF}^-$, $\mathbf{Z}^-$	ZF / Z without Foundation or Choice	CH-SET
$\mathrm{ord}(A,R)$	Order type of (A,R)	CH-SET
α^+	Ordinal successor	CH-SET
$\mathfrak{a}^{\oplus}$	Cardinal successor	CH-SET
$\mathfrak{a}$	Cardinal (fraktur)	CH-SET
V_α	Stage α of the cumulative hierarchy	CH-SET
$\mathrm{rank}(x)$	Rank of set x	CH-SET
$f[A]$	Image of A under f	CH-SET
$\mathrm{trcl}(A)$	Transitive closure of A	CH-SET

Further Reading

This text is compiled from the Open Logic Project (`openlogicproject.org`). The following sources provide deeper treatment of each domain.

General Logic Textbooks

- H. B. Enderton, *A Mathematical Introduction to Logic*, 2nd ed., Academic Press, 2001. Covers propositional and first-order logic through completeness with careful attention to detail.
- E. Mendelson, *Elements of Mathematical Logic*, 6th ed., CRC Press, 2015. A classic treatment including axiomatics, first-order logic, and incompleteness.
- D. Marker, *Model Theory: An Introduction*, Springer, 2002. Graduate-level model theory; excellent for the semantic and metatheoretic material of CH-SEM and CH-META.

Set Theory (CH-BST, CH-SET)

- P. R. Halmos, *Naive Set Theory*, Springer, 1960. The foundational informal set theory underlying CH-BST.
- K. Kunen, *Set Theory: An Introduction to Independence Proofs*, North-Holland, 1980. Standard graduate reference for the formal set theory of CH-SET.
- T. Jech, *Set Theory*, 3rd millennium ed., Springer, 2003. Comprehensive reference for ordinals, cardinals, and the cumulative hierarchy.

Syntax and Semantics (CH-SYN, CH-SEM)

- W. Hodges, *A Shorter Model Theory*, Cambridge University Press, 1997. Concise introduction to structures, satisfaction, and elementary equivalence.

- C. C. Chang and H. J. Keisler, *Model Theory*, 3rd ed., North-Holland, 1990. Definitive reference for Löwenheim–Skolem, compactness, and ultraproducts.

Deduction (CH-DED)

- A. S. Troelstra and H. Schwichtenberg, *Basic Proof Theory*, 2nd ed., Cambridge University Press, 2000. The standard reference for natural deduction and sequent calculus, including cut elimination.

- R. M. Smullyan, *First-Order Logic*, Dover, 1995 (reprint). The original systematic treatment of analytic tableaux.

- S. R. Buss (ed.), *Handbook of Proof Theory*, North-Holland, 1998. Comprehensive survey of all major proof systems.

Computability (CH-CMP)

- N. J. Cutland, *Computability: An Introduction to Recursive Function Theory*, Cambridge University Press, 1980. Clear development of primitive and general recursive functions, Turing machines, and undecidability.

- R. I. Soare, *Recursively Enumerable Sets and Degrees*, Springer, 1987. Advanced treatment of the computability theory underlying representability and Church's thesis.

- H. Rogers, *Theory of Recursive Functions and Effective Computability*, MIT Press, 1987. The classic reference for recursion theory.

Metatheory and Incompleteness (CH-META)

- G. S. Boolos, J. P. Burgess, and R. C. Jeffrey, *Computability and Logic*, 5th ed., Cambridge University Press, 2007. Excellent coverage of Gödel's theorems, representability, and Tarski's theorem.

- C. Smorynski, *Self-Reference and Modal Logic*, Springer, 1985. Deep treatment of the derivability conditions, Löb's theorem, and the provability logic GL.

- P. Lindström, "On extensions of elementary logic," *Theoria*, 35:1–11, 1969. The original paper on Lindström's theorem characterizing first-order logic.

The Open Logic Project

- Open Logic Project, *Open Logic: A Complete Text*, available at `openlogicproject.org`. The upstream source of all material in this volume, released under a Creative Commons license.

- The taxonomy and dependency analysis underlying this systematization are documented in the `taxonomy/` directory of the project repository.

Index