

Introduction to Mathematical Logic

Course Notes¹

March 1, 2024

Manuel Bodirsky, Institut für Algebra, TU Dresden

¹Disclaimer: this is a draft and probably contains many typos and mistakes.
Please report them to Manuel.Bodirsky@tu-dresden.de.

Contents

Chapter 1. Introduction	5
Outline	5
Chapter 2. Propositional Logic	7
2.1. Propositional Formulas	7
2.1.1. Syntax	7
2.1.2. Semantics	8
2.1.3. Disjunctive and conjunctive normal form	9
2.2. The Satisfiability Problem	10
2.2.1. Propositional Resolution	12
2.2.2. The DPLL Algorithm	13
Chapter 3. First-order Logic	15
3.1. First-order Structures	15
3.1.1. Signatures	15
3.1.2. Structures	15
3.1.3. Substructures and Extensions	16
3.1.4. Reducts and Expansions	16
3.1.5. Homomorphisms	16
3.2. Formulas, Sentences, Theories	17
3.2.1. Terms	18
3.2.2. Semantics of terms	18
3.2.3. Formulas and sentences	18
3.2.4. Semantics of formulas	19
3.2.5. First-order theories	20
Chapter 4. Set Theory	25
4.1. ZFC	25
4.1.1. Sets and classes	28
4.1.2. Partial orders and Zorn's lemma	28
4.2. Ordinals	29
4.2.1. Well-orders	29
4.2.2. Definition of ordinals	30
4.2.3. Successor and limit ordinals	32
4.2.4. Ordinal arithmetic	33
4.2.5. The well-ordering theorem	34
4.3. Cardinals	34
Chapter 5. The Completeness Theorem	39
5.1. Logical Axioms	39
5.2. Formal Proofs	41
5.3. Consistency	43
5.4. Henkin Theories	44

5.5. Compactness	47
Chapter 6. Computability	51
6.1. Primitive Recursion	51
6.1.1. Primitive recursive functions	52
6.1.2. Primitive recursive sets	52
6.1.3. Bounded μ -recursion	53
6.1.4. Gödel numbers	54
6.1.5. The Ackermann function	55
6.2. Recursive Functions	58
6.3. Recursively Enumerable Sets	59
6.4. Turing Computable Functions	61
6.4.1. Turing machines	61
6.4.2. Partial recursive functions are Turing computable	63
6.4.3. Turing computable functions are partial recursive	65
6.4.4. Universal machines and universal functions	67
6.4.5. The halting problem	69
Chapter 7. The Incompleteness Theorems	73
7.1. Coding Formulas and Proofs	73
7.2. Decidable Theories	74
7.3. (Weak) Peano Arithmetic	75
7.4. The Theorems of Tarski and Church	80
7.5. Gödel's First Incompleteness Theorem	81
7.6. Σ_1 -Definability of Truth of Σ_1 -Sentences	81
7.7. Gödel's Second Incompleteness Theorem	82
Chapter 8. Further Reading	85
Bibliography	89

CHAPTER 1

Introduction

Understanding and finding proofs is a central part of mathematics. But what *is* a proof, actually? Can the concept of a *proof* be formalised? Can we program a computer so that the computer can verify proofs and even search for proofs? What are the usual axioms of mathematics that we are allowed to use in proofs?

To formalise the notion of a proof we need logic; more specifically, we will work with so-called *first-order logic*. First-order logic is in many respects the most important logic, both in mathematics and in computer science. It strikes a good balance between *expressiveness* on the one hand and *good mathematical properties* on the other hand. For example, first-order logic is expressive enough to formulate the axioms of axiomatic set theory, e.g., Zermelo–Fraenkel set theory (ZFC; Chapter 4) or Bernays–Gödel set theory. Practically all of mathematics can be formalised in this way. What is even more important is that first-order logic has a very rich model theory; here we have to refer to model theory courses. In computer science, first-order logic may be viewed as the most important database query language. Various restrictions and extensions of first-order logic might be relevant in applications, but first-order logic remains the central point of departure.

The first bigger goal of the course is proving *Gödel’s completeness theorem* which shows that there exists a notion of formal proof such that a statement in first-order logic is true (valid) if and only if it has a formal proof. This statement has remarkable consequences; for example, it implies the *compactness theorem* for first-order logic, which has a great number of beautiful applications throughout mathematics, as we will see. The name *compactness* is borrowed from topology, and indeed there is a certain topological space such that the compactness theorem for first-order logic translates into the statement that this space is compact.

A remarkable feature of our notion of a formal proof is that formal proofs can be checked by a computer program. Therefore, proofs can also be found by a program: a program may exhaustively list all potential proofs and then check their correctness. Obviously, this is not an efficient way of searching for proofs. What appears to be a computer science topic actually has quite remarkable consequences in mathematics, because we can use this fact to prove that there are first-order sentences about set theory that are *independent from ZFC*, i.e., there are models of ZFC where the sentence is true, and there are models of ZFC where the sentence is false. In fact, such independent statements must exist for any formal system that can express a sufficiently strong (in a certain formal sense) part of arithmetic; this is *Gödel’s first incompleteness theorem*. This theorem can be strengthened as follows: we may even prove that if ZFC is consistent, then there is no proof in ZFC that ZFC is consistent; this is *Gödel’s second incompleteness theorem*.

Outline. Before introducing first-order logic, we start with an elementary chapter about propositional logic. This is a small logic microcosmos, and certain general ideas and basic facts of logic can already be introduced there (Chapter 2).

We then introduce first-order logic, in several steps: we first introduce (*first-order*) *structures*. These are the objects that first-order logic is talking about. I assume that the reader is already familiar with some examples of structures, such as fields, groups, graphs, or partial orders. However, I do not assume that the reader is familiar with the formal definition of structures and present the formal definition in full detail, including basic operations on structures such as *homomorphisms*, *products*, *extensions*, *substructures*, *expansions*, and *reducts* of general structures. We then introduce the *syntax* of first-order logic (i.e., how do first-order formulas look like?) and finally the *semantics* of first-order logic (i.e., what is the *meaning* of a first-order formula?). Then central definition here is the definition of a *model* of a first-order sentence or a first-order theory (which is simply a set of first-order sentences). We also discuss plenty of examples of structures, sentences, theories, and their models (Chapter 3).

To illustrate the power of first-order logic, we then present the first-order theory of the Zermelo–Fraenkel axioms of set theory (ZF) which might be extended by the axiom of choice (ZFC). Within ZFC, we may then formally introduce ordinal and cardinals (Chapter 4).

The concept of a *formal proof*, introduced in Chapter 5, consists of two parts: one part consists of a set of easy-to-check *logical axioms*; it will be easy to see that they are *valid*, that is, every structure is a model of these axioms. The second part is a description of how to deduce new valid sentences from sentences that are already known to be valid. The key property of the notion of a formal proof is that *all* statements that are valid can be deduced with the system, and that the deduction steps are simple so that they can be performed by a computer. Also in Chapter 5 we present an important consequence of the completeness theorem, namely the compactness theorem.

Chapters 6 and 7 explore the limits of the formal method. We first introduce μ -recursion and prove that these are precisely the functions that can be computed by a Turing machine. There are problems that cannot be solved by a Turing machine; the so-called *Halting problem* is one of them. It will be an easy consequence that Peano arithmetic is incomplete. The proof of Gödel’s first incompleteness theorem (in a strengthening due to Rosser) and then Gödel’s second incompleteness theorem requires further work.

We assume familiarity with basic (*‘naive’*) set theory. The set $\{0, 1, 2, \dots\}$ of natural numbers is denoted by \mathbb{N} .

The text contains 124 exercises; the ones with a star are harder.

Propositional Logic

Propositional logic can be seen as a part of first-order logic, but is much simpler and serves as a gentle introduction to the subject.

2.1. Propositional Formulas

A basic principle in logic is the separation of *syntax* and *semantics*. The syntax of a logic specifies how we may combine logical symbols to arrive at well-formed expressions. The *semantics* of a logic specifies how to assign a meaning to these expressions.

2.1.1. Syntax. The symbols of propositional logic are the symbols \wedge (*conjunction*, pronounced *and*), \neg (*negation*, pronounced *not*), \top (*true*), the brackets ‘(’ and ‘)’, and variable symbols. Typically we will use X, Y, Z , or X_1, X_2, \dots as variable symbols, but the choice of the variable symbols does not matter (except that they should be distinct from the other logical symbols). The syntax of propositional logic is defined recursively. We define *propositional formulas* as follows.

- (1) \top is a propositional formula;
- (2) all variables are propositional formulas;
- (3) if ϕ is a propositional formula, then $\neg\phi$ is also a propositional formula.
- (4) if ϕ_1 and ϕ_2 are propositional formulas, then $(\phi_1 \wedge \phi_2)$ is a propositional formula.

For example,

$$(\neg(X \wedge Y) \wedge Z) \tag{1}$$

is a propositional formula over the variables X, Y, Z . The outermost brackets may be omitted; that is, we may write $\neg(X \wedge Y) \wedge Z$ for the formula in (1). It is easy to see (see Exercise 1) that every propositional formula is of precisely *one* of the forms (1)-(4) in the recursive definition of propositional formulas. So we can associate to each propositional formula a unique *syntax tree* (which typically is depicted to grow from top to bottom); we do not go further into details, but rather illustrate the syntax tree of (1):

$$\begin{array}{c}
 (\neg(X \wedge Y) \wedge Z) \\
 \wedge \\
 \neg(X \wedge Y) \quad Z \\
 | \\
 (X \wedge Y) \\
 \wedge \\
 X \quad Y
 \end{array}$$

Exercises.

- (1) Show that every propositional formula is of precisely *one* of the forms (1)-(4) in the recursive definition of propositional formulas.

Hint: first prove by induction on the length of formulas that no proper initial segment of a formula is a formula.

2.1.2. Semantics. Let ϕ be a propositional formula with the variables X_1, X_2, \dots, X_n . A (*Boolean*) *assignment* for X_1, \dots, X_n is a function $s: \{X_1, \dots, X_n\} \rightarrow \{0, 1\}$.

REMARK 2.1.1. Here, 0 stands for *false* and 1 stands for *true*. We say that s *satisfies* ϕ if, informally, ϕ ‘evaluates to true’ if we replace X_i , for $i \in \{1, \dots, n\}$, by $s(X_i)$. Of course, all of this needs to be defined formally, and what I wrote in this paragraph is just to guide you when reading the formal definition below.

Formally, we define the satisfaction relation inductively over the structure of propositional formulas.

- (1) s satisfies \top (so \top stands for *true*).
- (2) s satisfies a variable X_i , for $i \in \{1, \dots, n\}$, if $s(X_i) = 1$.
- (3) s satisfies $\neg\phi$ if s does *not* satisfy ϕ .
- (4) s satisfies $(\phi_1 \wedge \phi_2)$ if s satisfies ϕ_1 and if s satisfies ϕ_2 .

Let ϕ be a propositional formula with variables X_1, \dots, X_n . Then ϕ is called

- *satisfiable* if there exists an assignment $s: \{X_1, \dots, X_n\} \rightarrow \{0, 1\}$ that satisfies ϕ , and *unsatisfiable* otherwise.
- a *tautology* (or *tautological*) if every assignment $s: \{X_1, \dots, X_n\} \rightarrow \{0, 1\}$ satisfies ϕ .

Clearly, ϕ is unsatisfiable if and only if $\neg\phi$ is a tautology.

Propositional formulas can be used to describe Boolean operations. Recall that an *operation of arity* $n \in \mathbb{N}$ *over a set* A is a function from A^n to A . A *Boolean operation* is an operation over $\{0, 1\}$. Let ϕ be a propositional formula with variables X_1, \dots, X_n . Then ϕ describes the Boolean operation that maps $(a_1, \dots, a_n) \in \{0, 1\}^n$ to 1 if the map $X_i \mapsto a_i$ satisfies ϕ , and to 0 otherwise. Note that for this definition it is important that we consider a fixed tuple of variables (so that we know which argument of the Boolean operation corresponds to which variable).

We use \perp as a shortcut for $\neg\top$ (which stands for *false*). If ϕ_1 and ϕ_2 are propositional formulas, then $\phi_1 \vee \phi_2$ (*disjunction*) is a shortcut for $\neg(\neg\phi_1 \wedge \neg\phi_2)$. The Boolean operation described by the propositional formulas $X_1 \wedge X_2$ and $X_1 \vee X_2$ are given by the following tables.

\wedge	0	1
0	0	0
1	0	1

\vee	0	1
0	0	1
1	1	1

There are other important shortcuts.

- $\phi_1 \Rightarrow \phi_2$ (*implication*) is a shortcut for $\neg\phi_1 \vee \phi_2$. We may read $\phi_1 \Rightarrow \phi_2$ as ‘if ϕ_1 (*is true*), then ϕ_2 (*is true*)’.
- $\phi_1 \Leftrightarrow \phi_2$ (*equivalence*) is a shortcut for $((\phi_1 \Rightarrow \phi_2) \wedge (\phi_2 \Rightarrow \phi_1))$. We may read $\phi_1 \Leftrightarrow \phi_2$ as ‘ ϕ_1 (*is true*) if and only if ϕ_2 (*is true*)’.
- $\phi_1 \oplus \phi_2$ (*exclusive or*) is a shortcut for $(\phi_1 \vee \phi_2) \wedge \neg(\phi_1 \wedge \phi_2)$.

The Boolean operations described by $X_1 \Rightarrow X_2$, by $X_1 \Leftrightarrow X_2$, and by $X_1 \oplus X_2$ can also be described by the following tables.

\Rightarrow	0	1
0	1	1
1	0	1

\Leftrightarrow	0	1
0	1	0
1	0	1

\oplus	0	1
0	0	1
1	1	0

Let ϕ_1 and ϕ_2 be propositional formulas with variables X_1, \dots, X_n . Then ϕ_1 and ϕ_2 are called *equivalent* if they describe the same Boolean operation. For example, note that for all propositional formulas ϕ_1, ϕ_2, ϕ_3 , the formulas $(\phi_1 \wedge (\phi_2 \wedge \phi_3))$ and $((\phi_1 \wedge \phi_2) \wedge \phi_3)$ are equivalent. Since we usually only care about the Boolean operation that is described by a propositional formula, we sometimes omit the brackets and

simply write $\phi_1 \wedge \phi_2 \wedge \phi_3$ instead of the propositional formulas above. If $I = \{i_1, \dots, i_n\}$ is a finite set and ϕ_i is a propositional formula for every $i \in I$, then we write $\bigwedge_{i \in I} \phi_i$ instead of $\phi_{i_1} \wedge \dots \wedge \phi_{i_n}$. It will be convenient to also use this expression if $n = 0$ (i.e., for $I = \emptyset$), in which case the expression equals \top by definition. Similarly, we write $\bigvee_{i \in I} \phi_i$ instead of $\phi_{i_1} \vee \dots \vee \phi_{i_n}$; the empty disjunction equals \perp by definition.

Exercises.

- (2) Let ϕ_1 and ϕ_2 be propositional formulas with variables X_1, \dots, X_n . Show that ϕ_1 and ϕ_2 are equivalent if and only if $\phi_1 \Leftrightarrow \phi_2$ is a tautology.
 (3) Show that the following propositional formula is a tautology.

$$(X \Rightarrow Y) \Leftrightarrow (\neg Y \Rightarrow \neg X) \quad (\text{Contraposition})$$

- (4) There are three doors to different rooms; we know that there is a treasure in one of the rooms and a dragon in each of the other two.
- The first door is labelled ‘A dragon is in this room’.
 - The second door is labelled ‘A treasure is in this room’.
 - The third door is labelled ‘A dragon is in the second room’.

We also know that at most one of the three labels is true.

Task: Model this problem using a propositional formula. Explain the intended meaning for the propositional variables that you are using, and how you translate the given information into a formula. Which room contains the treasure?

2.1.3. Disjunctive and conjunctive normal form. Every propositional formula describes a Boolean operation; conversely, every Boolean operation can be described by a propositional formula.

PROPOSITION 2.1.2. *Every Boolean operation $f: \{0, 1\}^n \rightarrow \{0, 1\}$ can be described by a propositional formula with variables X_1, \dots, X_n .*

PROOF. For all $a_1, \dots, a_n \in \{0, 1\}$ we have that $f(a_1, \dots, a_n) = 1$ if and only if the assignment $X_i \mapsto a_i$ satisfies

$$\bigvee_{\substack{b_1, \dots, b_n \in \{0, 1\}, \\ f(b_1, \dots, b_n) = 1}} \left(\bigwedge_{i \in \{1, \dots, n\} \text{ with } b_i = 1} X_i \wedge \bigwedge_{i \in \{1, \dots, n\} \text{ with } b_i = 0} \neg X_i \right). \quad \square$$

EXAMPLE 1. Let $f: \{0, 1\}^3 \rightarrow \{0, 1\}$ be the Boolean operation that is given by the table on the right.

a_1	0	0	0	0	1	1	1	1
a_2	0	0	1	1	0	0	1	1
a_3	0	1	0	1	0	1	0	1
$f(a_1, a_2, a_3)$	0	1	0	1	0	0	0	0

Then f is described by the propositional formula

$$(\neg X_1 \wedge \neg X_2 \wedge X_3) \vee (\neg X_1 \wedge X_2 \wedge X_3). \quad \triangle$$

The proof of Proposition 2.1.2 shows something stronger: we see that every Boolean operation can be described by a disjunction of conjunctions of variables or negated variables. This specific form of a propositional formula is called *disjunctive normal form (DNF)*.

It can also be shown that every Boolean operation $f: \{0, 1\}^n \rightarrow \{0, 1\}$ can be described by a propositional formula in *conjunctive normal form (CNF)*, which is a conjunction of disjunctions of variables or negated variables. To see this, we apply

Proposition 2.1.2 to the Boolean operation which returns 0 if f returns 1, and returns 1 if f returns 0. We already know that there exists a propositional formula ϕ in DNF that describes this new Boolean operation. Then clearly $\neg\phi$ describes f . Note that $\neg\phi$ is not yet in conjunctive normal form; but using the following equivalences and induction, one can easily find a formula which is in CNF and equivalent to $\neg\phi$.

$$\neg(\neg\phi) \Leftrightarrow \phi \quad (2)$$

$$\neg(\phi_1 \vee \phi_2) \Leftrightarrow (\neg\phi_1 \wedge \neg\phi_2) \quad (3)$$

$$\neg(\phi_1 \wedge \phi_2) \Leftrightarrow (\neg\phi_1 \vee \neg\phi_2) \quad (4)$$

The tautologies in (3) and in (4) are also called *De Morgan's laws*. The same idea can be used to transform CNF into DNF.

There is an alternative approach for converting between CNF and DNF, based on using the distributive laws given below (we leave the details to the reader)

$$\phi_1 \wedge (\phi_2 \vee \phi_3) \Leftrightarrow ((\phi_1 \wedge \phi_2) \vee (\phi_1 \wedge \phi_3)) \quad (\wedge\text{-Distributivity})$$

$$\phi_1 \vee (\phi_2 \wedge \phi_3) \Leftrightarrow ((\phi_1 \vee \phi_2) \wedge (\phi_1 \vee \phi_3)) \quad (\vee\text{-Distributivity}).$$

The CNF is particularly useful; often, propositional formulas are assumed to be in CNF. So there is specialised terminology: the conjuncts of a propositional formula in CNF are also called *clauses*. The disjuncts of a clause are called *literals*; they are either variables or negated variables. In the first case, the literal is called *positive* and in the latter case *negative* (correspondingly, we also speak about positive and negative occurrences of variables in clauses). A clause is viewed as a set of literals (here we do not lose essential information, because changing the order or the number of occurrences of the literals leads to equivalent formulas).

Exercises.

- (5) Find a propositional formula in CNF that describes the Boolean operation from Example 1.
 (6) Show that *every* propositional formula in CNF that is equivalent to

$$(X_1 \wedge Y_1) \vee (X_2 \wedge Y_2) \vee \cdots \vee (X_n \wedge Y_n)$$

has at least 2^n clauses.

- (7) Show that every Boolean operation $f: \{0, 1\}^n \rightarrow \{0, 1\}$ can be written in the form

$$(x_1, \dots, x_n) \mapsto \bigoplus_{i \in \{1, \dots, \ell\}} \bigwedge_{j \in S_i} x_j$$

where $S_1, \dots, S_\ell \subseteq \{1, \dots, n\}$.

- (8) Show that a Boolean operation $f: \{0, 1\}^n \rightarrow \{0, 1\}$ is *idempotent*, i.e., $f(x, \dots, x) = x$ for all $x \in \{0, 1\}$, if and only if f can be written in the form

$$(x_1, \dots, x_n) \mapsto \bigoplus_{i \in \{1, \dots, 2k+1\}} \bigwedge_{j \in S_i} x_j$$

where $S_1, \dots, S_{2k+1} \subseteq \{1, \dots, n\}$.

2.2. The Satisfiability Problem

The satisfiability problem is a computational problem, often denoted by *SAT*: for a given a proposition formula ϕ , the task is to determine whether ϕ is satisfiable. Many combinatorial problems can be coded into the satisfiability problem. Note that the problem of deciding whether a given propositional formula ϕ is a tautology can be reduced to the satisfiability problem: ϕ is a tautology if and only if $\neg\phi$ is unsatisfiable.

The satisfiability problem can be solved by a computer: we first list exhaustively all functions s from the variables X_1, \dots, X_n of ϕ to $\{0, 1\}$; for each of the functions we can verify easily whether s satisfies ϕ , by recursion on the syntax tree of ϕ from Section 2.1.1, evaluating the formula following the rules specified in Section 2.1.2. Unfortunately, the running time of the procedure is exponential, and the brute-force algorithm above is impractical already for inputs of moderate size. There is no algorithm known that can solve SAT in polynomial time in the size of the formula (i.e., the number of symbols of the formula).

REMARK 2.2.1. The naive algorithm for SAT that we have presented above shows that the satisfiability problem belongs to the complexity class NP; see [23] for an introduction to computational complexity.

EXAMPLE 2. Is there a graph G with 43 vertices that contains neither a *clique with 5 vertices* nor an *independent set with 5 vertices*? A *clique* in a graph is a subset S of the vertices such every vertex in S is adjacent in the graph to any other vertex in S . An *independent set* in a graph is a subset S of the vertices such that every vertex in S is not adjacent in the graph to any other vertex in S . This problem can be formulated as an instance of SAT as follows. We use the variables $X_{1,2}, X_{1,3}, \dots, X_{42,43}$ and consider the following propositional formula.

$$\bigwedge_{S \subseteq \{1, \dots, 43\}, |S|=5} \left(\bigvee_{i,j \in S, i \neq j} X_{i,j} \wedge \bigvee_{i,j \in S, i \neq j} \neg X_{i,j} \right)$$

If the given propositional formula is satisfiable, then the answer to the question above is *yes*. This question is a famous open problem in Ramsey theory. The algorithm for the satisfiability problem mentioned above needs to examine 2^n cases, where n is the number of variables; for the above problem, $n = 43 \cdot 42/2 = 903$. IBM is currently working on a 100 GHz CPU, that is, the CPU has 100 billion clock cycles per second. Even if we could solve one case in one clock cycle, the computer would need $2^{903}/(31536000 * 10^{11}) > 10^{\lg 2 \cdot 903}/10^{19} > 10^{0,301 \cdot 903 - 19} > 10^{252}$ years. For comparison: the age of the universe is estimated to be $13.79 \cdot 10^9$ years. \triangle

REMARK 2.2.2. The class of all problems that can be solved in polynomial time in the size of the input is denoted by P. So it is an open problem whether SAT is in P. It can be shown that the satisfiability problem belongs to the hardest problems in NP; it is *NP-complete*. If there is a polynomial-time algorithm for SAT, then P would be equal to NP; most researchers believe that P and NP are distinct. The P=NP question is one of the famous Millenium problems and believed to be one of the most difficult questions in mathematics.

PROPOSITION 2.2.3. *There exists an algorithm that computes for every propositional formula ϕ in polynomial time in the size of ϕ another propositional formula ψ in CNF such that ψ is satisfiable if and only if ϕ is satisfiable.*

PROOF IDEA. Suppose that ϕ is not yet in CNF; for example, suppose that ϕ is of the form $((\phi_1 \wedge \phi_2) \vee \phi_3) \wedge \phi_4$ for some propositional formulas $\phi_1, \phi_2, \phi_3, \phi_4$. We then introduce a *new* variable X , i.e., a variable that does not yet appear in ϕ_1, \dots, ϕ_4 , and define

$$\psi := (\phi_1 \vee X) \wedge (\phi_2 \vee X) \wedge (\phi_3 \vee \neg X) \wedge \phi_4.$$

The formulas ϕ and ψ are not equivalent, because they do not have the same set of variables. However, ψ is satisfiable if and only if ϕ is satisfiable. Repeatedly applying such transformations we eventually arrive at a formula in CNF. Moreover, the resulting formula is of linear size in the original formula ϕ , and can be easily computed from ϕ . \square

Exercises.

- (9) Show that there exists a polynomial-time algorithm that decides the satisfiability problem for propositional formulas given in DNF.
- (10) We have seen that every propositional formula is equivalent to a formula in DNF, and that the satisfiability problem for propositional formulas can be solved in polynomial time. Why doesn't this provide a solution to the Millenium problem about a polynomial-time algorithm for SAT?
- (11) Solve the exercises on the interactive site
<https://iltis.cs.tu-dortmund.de/Logic-WiSe2022-external/en/>
 in Chapter 3: Satisfiability of propositional formulas.

2.2.1. Propositional Resolution. Propositional resolution provides a conceptually simple algorithm to solve the satisfiability problem; however, similarly as the brute-force algorithm that we have seen in the previous section, it is not an efficient algorithm, neither in theory nor in practice.

To formulate propositional resolution, we assume that the propositional formula ϕ is given in conjunctive normal form, as a set of clauses. We now apply the following inference rule:

$$\frac{\{L_1, \dots, L_n, X\}, \quad \{\neg X, M_1, \dots, M_k\}}{\{L_1, \dots, L_n, M_1, \dots, M_k\}} \quad (5)$$

Such rules are to be read as follows: suppose ϕ contains the two clauses that are written above the line. Then we *add* the clause below the line to ϕ . Note that in order to apply the rule (5), one of the clauses must contain a variable X , and the other must contain the same variable, but negated, $\neg X$. Also note that the formula obtained from ϕ by adding the new clause is equivalent to ϕ : if s is a satisfying assignment for ϕ , and $s(X) = 1$, then s must satisfy the literal M_i for some $i \leq k$ because s satisfies the right clause above the line. If $s(X) = 0$, then s must satisfy the literal L_i for some $i \leq n$ because s satisfies the left clause above the line. In both cases it follows that s satisfies the clause below the line.

Note that if we apply the resolution rule exhaustively to derive new clauses, we eventually arrive at a set of clauses where no new clauses can be derived, because for a fixed finite set of variables there are only finitely many clauses that can be formed. Such sets of clauses will be called *closed under resolution*. If ϕ contains an empty clause, then clearly ϕ is unsatisfiable (this property of resolution is sometimes referred to as *soundness* of resolution¹) The following lemma states a converse to this fact (this property is sometimes referred to as *completeness* of resolution).

THEOREM 2.2.4. *Let ϕ be a propositional formula in CNF which is closed under resolution, and does not contain the empty clause. Then ϕ is satisfiable.*

To prove the theorem we need some preparations. If ϕ is a propositional formula in CNF and L a literal of ϕ , then we write ϕ_L for the formula obtained from ϕ by dropping all clauses that contain L and removing all occurrences of the literal $\neg L$ in all the remaining clauses (here and in the following we identify $\neg(\neg L)$ with L).

LEMMA 2.2.5. *Let ϕ be a formula in CNF that is closed under resolution and let L be a literal of ϕ . Then ϕ_L is closed under resolution, too.*

PROOF. Let C_1 and C_2 be two clauses in ϕ_L that resolve to C . We have to show that C is a clause of ϕ_L . Note that neither C_1 nor C_2 contains the literal $\neg L$, so

¹Soundness is often translated to *Korrektheit* in German, which is also the translation of *correctness*. However, correctness is often used as soundness *and* completeness. We therefore propose to translate soundness with *Stimmigkeit*.

neither does C . If C_1 and C_2 were already clauses of ϕ , then so is C , and hence C is a clause of ϕ_L . Otherwise, C_1 was obtained from the clause $C_1 \cup \{\neg L\}$ of ϕ or C_2 was obtained from the clause $C_2 \cup \{\neg L\}$ of ϕ . Since ϕ is closed under resolution it contains $C \cup \{\neg L\}$, and hence ϕ_L contains the clause C . \square

PROOF OF THEOREM 2.2.4. We prove the statement by induction on the number of variables that appear in ϕ . First suppose that ϕ contains a clause that consists of a single negative literal $\{\neg X\}$. Then $\phi_{\neg X}$ cannot contain the empty clause: otherwise, ϕ must contain the clause $\{X\}$, and we could have derived the empty clause from the clauses $\{X\}$ and $\{\neg X\}$ by resolution, contradicting the assumptions that ϕ is closed under resolution and does not contain the empty clause. By Lemma 2.2.5, the formula $\phi_{\neg X}$ is closed under resolution, but has less variables, so by the inductive assumption it has a satisfying assignment s . Then the extension of s given by $X \mapsto 0$ is a satisfying assignment to the original formula.

So we assume in the following that ϕ does not contain clauses of the form $\{\neg X\}$. If all variables appear negatively in ϕ , then the function that maps all variables to 0 is a satisfying assignment and we are done. So ϕ contains a clause with a positive literal, say X . Then ϕ_X does not contain the empty clause because of the assumption that ϕ does not contain clauses of the form $\neg X$. Moreover, ϕ_X is closed under resolution by Lemma 2.2.5, and has fewer variables, so it has a satisfying assignment s by the inductive assumption. Then the extension of s defined by $X \mapsto 1$ is a satisfying assignment to the original formula. \square

Exercises.

- (12) Find for every n a propositional formula ϕ in CNF with n variables and polynomially many clauses in n such that the closure of ϕ under resolution has 2^n many clauses.
- (13) A propositional formula in CNF is called *bijunctive* if all clauses have at most two literals. Show that resolution can be used to obtain a polynomial-time algorithm for the satisfiability problem for bijunctive formulas.
- (14) A propositional formula in CNF is called *Horn* if each clause of the formula contains at most one positive literal. Show that there is a solution to Exercise 12 where ϕ is a Horn formula.
- (15) Show that the worst-case running time of resolution is exponential even if the input is restricted to Horn formulas.

2.2.2. The DPLL Algorithm. In this section we present a third algorithm for SAT, the Davis-Putnam-Logemann-Loveland (DPLL) algorithm, which is the basis for almost all modern SAT solvers. Such SAT solvers have reached remarkably good performance on randomly generated instances of SAT and on large instances of SAT that arise in industry (but they do not perform sufficiently well on difficult hand-crafted instances as the one in Example 2). In essence, the DPLL algorithm is closely related to propositional resolution. Again we assume that the propositional formula ϕ is given in conjunctive normal form. We use an important inference step, called *unit propagation* (which you might recognise from the completeness proof of the resolution procedure).

Unit Propagation. If ϕ contains a clause of the form $\{X\}$, i.e., a clause that just contains one positive literal, then X must take value 1 in every solution to ϕ . Hence, all clauses that contain the literal X can be removed, and all literals of the form $\neg X$ can be removed from the remaining clauses. In this way we have eliminated the variable X . Similarly, unit propagation eliminates X if ϕ contains a clause of the form $\{\neg X\}$.

Pure Literal Elimination. If a variable X only occurs positively in ϕ , then we may remove all clauses that contain X , and solve the resulting instance recursively. If the resulting instance has a satisfying assignment, we obtain a satisfying assignment to ϕ extending α by $X \mapsto 1$. Similarly, we can simplify the instance if X only occurs negatively in ϕ (in these cases we say that X is *pure* in ϕ).

The overall algorithm can be found in Figure 2.1.

```

DPLL( $\phi$ )
// Input: A propositional formula  $\phi$  in CNF.
// Output: yes if  $\phi$  is satisfiable, no otherwise.
While there are unit clauses or pure literals in  $\phi$ :
    apply unit propagation and pure literal elimination.
If  $\phi$  contains an empty clause then return no.
If  $\phi$  contains no clauses then return yes.
Pick a literal  $L$  from some clause.
Return yes if DPLL( $\phi \cup \{\neg L\}$ ) returns yes or DPLL( $\phi \cup \{L\}$ ) returns yes,
otherwise return no.

```

FIGURE 2.1. An algorithm for the satisfiability problem.

Exercises.

- (16) Show that repeatedly applying Unit Propagation is a sound and complete procedure to test satisfiability of propositional Horn formulas (see Exercise 15). Show that the running time of this algorithm is polynomial. Bonus: Find an implementation of this algorithm with linear running time.
- (17) Show that repeatedly applying Unit Propagation is not complete to test satisfiability of bijunctive formulas (see Exercise 15).

First-order Logic

In this section we introduce *first-order logic*. First-order logic can be seen as an extension of propositional logic by the ability to talk about functions and relations of so-called *structures* (sometimes also called *first-order structures*) and in particular the ability to *quantify* over the elements of the structure. There are many other logics: as the name suggests, there is for instance *second-order logic*, where it is also allowed to quantify over subsets of elements of the domain, or even over relations.

We first introduce structures (Section 3.1). Then we introduce the syntax and semantics of first-order logic. The syntax of first-order logic has two ‘layers’: we first have to introduce terms (Section 3.2.1), and then formulas (Section 3.2.1).

3.1. First-order Structures

3.1.1. Signatures. A *signature* τ is a set of relation and function symbols, each equipped with an *arity* $k \in \mathbb{N}$. Examples of important signatures:

- $\tau_{\text{Graph}} = \{E\}$
- $\tau_{\text{AGroup}} = \{+, -, 0\}$
- $\tau_{\text{Ring}} = \tau_{\text{AG}} \cup \{1, \cdot\}$
- $\tau_{\text{Group}} = \{\circ, ^{-1}, e\}$
- $\tau_{\text{LO}} = \{<\}$
- $\tau_{\text{OGroup}} = \tau_{\text{Group}} \cup \tau_{\text{LO}}$
- $\tau_{\text{Arithm}} = \{0, s, +, \cdot, <\}$
- $\tau_{\text{Set}} = \{\in\}$

Here, $+$, \circ , and \cdot are binary function symbols, $-$, $^{-1}$, and s are unary function symbols, e , 0 , and 1 are 0-ary function symbols, and E , \in , and $<$ are binary relation symbols.

3.1.2. Structures. A τ -*structure* \underline{A} is a non-empty¹ set A (called the *domain*, or the *base set*, or the *universe* of \underline{A}) together with

- a relation $R^{\underline{A}} \subseteq A^k$ for each k -ary relation symbol $R \in \tau$. Here we allow the case $k = 0$, in which case $R^{\underline{A}}$ is either empty or of the form $\{()\}$, i.e. the set consisting of the empty tuple;
- a function $f^{\underline{A}}: A^k \rightarrow A$ for each k -ary function symbol $f \in \tau$; here we also allow the case $k = 0$ to model constants from A .

Unless stated otherwise, A, B, C, \dots denotes the domain of the structure $\underline{A}, \underline{B}, \underline{C}, \dots$, respectively. We sometimes write $(A; R_1^{\underline{A}}, R_2^{\underline{A}}, \dots, f_1^{\underline{A}}, f_2^{\underline{A}}, \dots)$ for the structure \underline{A} with relations $R_1^{\underline{A}}, R_2^{\underline{A}}, \dots$ and functions $f_1^{\underline{A}}, f_2^{\underline{A}}, \dots$. Well-known examples of structures are $(\mathbb{Q}; +, -, 0, 1, \cdot, <)$, $(\mathbb{C}; +, -, 0, 1, \cdot, <)$, $(\mathbb{Z}; 0, s, +, \cdot, <)$, etc. In cases where the reference to the structure \underline{A} is clear we are sometimes sloppy with the notation and write f instead of $f^{\underline{A}}$. We say that a structure is infinite if its domain is infinite.

¹In some texts, structures with empty sets are allowed. Occasionally, this makes a difference, but it is always straightforward to translate between results in one setting to results in the other setting.

EXAMPLE 3. A (*simple, undirected*) graph is a pair (V, E) consisting of a set of vertices V and a set of edges $E \subseteq \binom{V}{2}$, that is, E is a set of 2-element subsets of V . Graphs can be modelled as relational structures \underline{G} using a signature that contains a single binary relation symbol R , putting $G := V$ and adding (u, v) to $R^{\underline{G}}$ if $\{u, v\} \in E$. If we insist that a structure with this signature satisfies $(x, y) \in R^{\underline{G}} \Rightarrow (y, x) \in R^{\underline{G}}$ and $(x, x) \notin R^{\underline{G}}$, then we can associate to such a structure an undirected graph and obtain a bijective correspondence between undirected graphs with vertices V and structures \underline{G} with domain V as described above. \triangle

EXAMPLE 4. A *group* is a structure \underline{G} with the signature $\tau_{\text{Group}} = \{\circ, ^{-1}, e\}$ such that for all $x, y, z \in G$ we have that

- $x \circ (y \circ z) = (x \circ y) \circ z$,
- $e \circ x = x \circ e = x$,
- $x \circ x^{-1} = x^{-1} \circ x = e$.

A group is Abelian if it additionally satisfies $x \circ y = y \circ x$ for all $x, y \in G$. For Abelian groups, one often uses the signature $\tau_{\text{AGroup}} = \{+, -, 0\}$ instead of τ_{Group} . \triangle

3.1.3. Substructures and Extensions. A τ -structure \underline{A} is a *substructure* of a τ -structure \underline{B} iff

- $A \subseteq B$,
- for each $R \in \tau$, and for all tuples \bar{a} from A , $\bar{a} \in R^{\underline{A}}$ iff $\bar{a} \in R^{\underline{B}}$, and
- for each $f \in \tau$ we have that $f^{\underline{A}}(\bar{a}) = f^{\underline{B}}(\bar{a})$.

In this case, we also say that \underline{B} is an *extension* of \underline{A} . Substructures \underline{A} of \underline{B} and extensions \underline{B} of \underline{A} are called *proper* if the domains of \underline{A} and \underline{B} are distinct.

Note that for every subset S of the domain of \underline{B} there is a unique smallest substructure of \underline{B} whose domain contains S , which is called the *substructure of \underline{B} generated by S* , and which is denoted by $\underline{B}[S]$.

EXAMPLE 5. When we view a graph as an $\{E\}$ -structure \underline{G} , then a subgraph is not necessarily a substructure of \underline{G} . In graph theory, the substructures of \underline{G} are called *induced subgraphs*: the difference is that in an induced subgraph (V', E') of (V, E) the edge set must be of the form $E' := E \cap \binom{V'}{2}$ instead of an arbitrary subset of it. \triangle

EXAMPLE 6. Due to the choice of our signature τ_{Group} , the subgroups of \underline{G} are precisely the substructures of \underline{G} as defined above. \triangle

3.1.4. Reducts and Expansions. Let σ, τ be signatures with $\sigma \subseteq \tau$. If \underline{A} is a σ -structure and \underline{B} is a τ -structure, both with the same domain, such that $R^{\underline{A}} = R^{\underline{B}}$ for all relations $R \in \sigma$ and $f^{\underline{A}} = f^{\underline{B}}$ for all function symbols $f \in \sigma$, then \underline{A} is called a (σ -) *reduct* of \underline{B} , and \underline{B} is called an *expansion* of \underline{A} .

EXAMPLE 7. A *linearly ordered group* is a τ_{OGroup} -structure \underline{G} whose τ_{Group} -reduct is a group and such that $<^{\underline{G}}$ is a linear order which is *translation-invariant*, i.e., $a \leq b$ implies $c \circ a \leq c \circ b$ for all $a, b, c \in G$, where $x \leq y$ stands for $x <^{\underline{G}} y$ or $x = y$. \triangle

DEFINITION 3.1.1. Let \underline{A} be a τ -structure and $B \subseteq A$. We write \underline{A}_B for the expansion of \underline{A} with the signature $\tau \cup \{c_b \mid b \in B\}$, where c_b is a new constant symbol for every element $b \in B$, such that $c_b^{\underline{A}_B} := b$.

3.1.5. Homomorphisms. In the following, let \underline{A} and \underline{B} be τ -structures. A *homomorphism* h from \underline{A} to \underline{B} is a mapping from A to B that *preserves* each function and each relation for the symbols in τ ; that is,

- if (a_1, \dots, a_k) is in $R^{\underline{A}}$, then $(h(a_1), \dots, h(a_k))$ must be in $R^{\underline{B}}$;

- $f^B(h(a_1), \dots, h(a_k)) = h(f^A(a_1, \dots, a_k))$.

A homomorphism from \underline{A} to \underline{B} is called a *strong homomorphism* if it also preserves the complements of the relations from \underline{A} . Injective strong homomorphisms are called *embeddings*. Surjective embeddings are called *isomorphisms*.

EXAMPLE 8. Group-, ring-, and field homomorphisms. \triangle

EXAMPLE 9. The *graph colorability problem* is an important problem in discrete mathematics, with many applications in theoretical computer science (it can be used to model e.g. frequency assignment problems). As a computational problem, the graph n -colorability problem has the following form.

Given: a finite graph $G = (V, E)$.

Question: can we colour the vertices of G with n colours such that adjacent vertices get different colours?

The n -colorability problem can be formulated as a graph homomorphism problem: is there a homomorphism from G to

$$K_n := (\{1, \dots, n\}; E^{K_n}) \text{ where } E^{K_n} := \{(u, v) \in \{1, \dots, n\}^2 \mid u \neq v\}.$$

We also refer to these homomorphisms as *proper colourings* of G , and say that G is *n -colourable* if such a colouring exists. The *chromatic number* $\chi(G)$ of G is the minimal natural number $n \in \mathbb{N}$ such that G is n -colourable. For example, the chromatic number of K_n is n . \triangle

EXAMPLE 10. We present a concrete instance of a mathematical colouring problem. Let (V, E) the unit distance graph on \mathbb{R}^2 , i.e., the graph has the vertex set $V := \mathbb{R}^2$ (we imagine the nodes as the points of the Euclidean plane) and edge set

$$E := \{(x, y) \in V^2 \mid |x - y| = 1\}.$$

In other words, two points are linked by an edge if they have distance one. What is the chromatic number of this graph? \triangle

We will see later (in Corollary 5.5.5, as a consequence of the compactness theorem) that a graph G is k -colourable if and only if all finite subgraphs of G are k -colourable. The problem to determine the chromatic number χ of the graph in Example 10 is known as the Hadwiger-Nelson problem. It is known that $\chi \leq 7$. We have seen that $4 \leq \chi$. In April 11, 2018, Aubrey de Grey announced a proof that $5 \leq \chi$. The precise value of $\chi \in \{5, 6, 7\}$ is not known.

Exercises.

- (18) Find a signature τ for vector spaces and describe how a vector space may be viewed as a τ -structure. Your definition should have the property that homomorphisms between the structures you consider correspond precisely to linear maps.
- (19) Let \underline{A} and \underline{B} be τ -structures and suppose that $G \subseteq A$ generates \underline{A} , i.e., $\underline{A} = \underline{A}[G]$. Then every homomorphism $h: \underline{A} \rightarrow \underline{B}$ is determined by its values on G .
- (20) Let \underline{A} and \underline{B} be τ -structures, and suppose that τ has no relation symbols. Show that every bijective homomorphism from \underline{A} to \underline{B} is an isomorphism.
- (21) Find an example of a τ -structure \underline{A} and a bijective homomorphism from \underline{A} to \underline{A} which is not an isomorphism.

3.2. Formulas, Sentences, Theories

To define the syntax of first-order logic, we first introduce terms, then (first-order) formulas and (first-order) sentences, and finally (first-order) theories.

3.2.1. Terms. Let τ be a signature. In this section we will see how to use the function symbols in τ to build *terms*.

DEFINITION 3.2.1. A (τ -)term is defined inductively:

- (1) constants from τ are τ -terms;
- (2) variables x_0, x_1, \dots are τ -terms;
- (3) if t_1, \dots, t_k are τ -terms, and $f \in \tau$ has arity k , then $f(t_1, \dots, t_k)$ is a τ -term.

Note that item (1) is a special case of item (3). If t is term, then we write $t(x_1, \dots, x_n)$ if all variables that appear in t come from $\{x_1, \dots, x_n\}$; we do not require that each variable x_i appears in t , but every variables that appears in t must be of the form x_i for some $i \leq n$.

EXAMPLE 11. Well-known examples of terms are polynomials over a ring R : they are terms over a signature that contains a binary symbols $+, \cdot$ for addition and multiplication, together with a constant symbol for each element of R . However, note that for $+$ and \cdot we usually use in-fix notation, i.e., we write $t_1 + t_2$ instead of $+(t_1, t_2)$. \triangle

EXAMPLE 12. Propositional formulas can be viewed as $\{\wedge, \neg, \top\}$ -terms. \triangle

3.2.2. Semantics of terms. In this section we describe how τ -terms over a given τ -structure describe functions (in the same way as polynomials describe polynomial functions over a given ring).

Let \underline{A} be a τ -structure and let x_1, \dots, x_n be distinct variables. Every τ -term $t(x_1, \dots, x_n)$ describes an operation $t^{\underline{A}}: A^n \rightarrow A$ as follows:

- (1) if t equals $c \in \tau$ then $t^{\underline{A}}$ is the operation $(a_1, \dots, a_n) \mapsto c^{\underline{A}}$;
- (2) if t equals x_i then $t^{\underline{A}}$ is the operation $(a_1, \dots, a_n) \mapsto a_i$;
- (3) if t equals $f(t_1, \dots, t_k)$ for a k -ary $f \in \tau$ and τ -terms t_1, \dots, t_k , then $t^{\underline{A}}$ is the operation

$$(a_1, \dots, a_n) \mapsto f^{\underline{A}}(t_1^{\underline{A}}(a_1, \dots, a_n), \dots, t_k^{\underline{A}}(a_1, \dots, a_n)).$$

The function described by t is also called the *term function* of t (with respect to \underline{A}).

Exercises.

- (22) Let \underline{B} be a τ -structure and $G \subseteq B$. Let \underline{A} be the substructure of \underline{B} generated by G . Show that $b \in B$ is an element of A if and only if there are elements $g_1, \dots, g_n \in G$ and a τ -term $t(x_1, \dots, x_n)$ such that $t^{\underline{B}}(g_1, \dots, g_n) = b$.

3.2.3. Formulas and sentences. Let τ be a signature. The relation symbols in the signature τ did not play any role when defining τ -terms, but they become important when defining τ -formulas. Moreover, the equality symbol $=$ is ‘hard-wired’ into first-order logic; we can use it to create formulas by equating terms. Finally, we can combine formulas using Boolean connectives, and quantify over variables.

DEFINITION 3.2.2. An *atomic τ -formula* is an expression of the form

- \top ;
- $t_1 = t_2$ where t_1 and t_2 are τ -terms;
- $R(t_1, \dots, t_k)$ where t_1, \dots, t_k are τ -terms and $R \in \tau$ is a k -ary relation symbol.

Then τ -formulas are defined inductively as follows:

- atomic τ -formulas are τ -formulas;
- if ϕ is a τ -formula, then $\neg\phi$ is a τ -formula (*negation*);
- if ϕ and ψ are τ -formulas, then $\phi \wedge \psi$ is a τ -formula (*conjunction*);

- if ϕ is a τ -formula, and x is a variable, then $\exists x. \phi$ is a τ -formula (*existential quantification*).

Note that we may reconstruct from every τ -formula, viewed as a string of symbols, the way how it was built inductively. The purpose of the dot in the formula $\exists x. \phi$ is to increase readability, e.g. in the formula $\exists na. \text{Nu}(na)$, which would otherwise become $\exists na \text{Nu}(na)$. In formulas of the form $\exists x. (\phi \wedge \psi)$ we omit the dot since this does not harm readability.

Atomic formulas and negations of atomic formulas are sometimes called *literals*. Similarly as for terms, we write $\phi(x_1, \dots, x_n)$ if all non-quantified variables in the formula ϕ come from $\{x_1, \dots, x_n\}$. A (*first-order*) *sentence* is a formula without free variables, i.e., all variables that appear in the formula are *bound*, i.e., quantified by some quantifier.

Shortcuts: We use the shortcuts \perp , \vee , \Rightarrow , \Leftrightarrow that we already know from propositional logic. Additionally, we define the following shortcuts:

- *Universal quantification:* $\forall x. \phi(x)$ is an abbreviation for $\neg \exists x. \neg \phi(x)$
- *Inequality:* $x \neq y$ is an abbreviation for $\neg(x = y)$

If A is a unary relation symbol, then we may write $\exists x \in A. \phi$ instead of $\exists x (A(x) \wedge \phi)$ and $\forall x \in A. \phi$ instead of $\forall x (A(x) \Rightarrow \phi)$.

EXAMPLE 13. Let $\tau = \{R\}$ where R is a binary relation symbol. Then the following formula is an example of a first-order sentence

$$\forall x_1, x_2, x_3 ((R(x_1, x_2) \wedge R(x_2, x_3)) \Rightarrow R(x_1, x_3)). \quad \triangle$$

3.2.4. Semantics of formulas. So far, we have just introduced the *syntax* of first-order logic, i.e., the shape of formulas and sentences, without discussing what these expressions actually mean. In this section we discuss their *semantics*; the idea is that formulas can be used to define new relations and functions in a given structure, and that sentences can be used to describe properties that a structure might have.

Let \underline{A} be a τ -structure. Then every τ -formula $\phi(x_1, \dots, x_n)$ describes a relation $\phi^{\underline{A}} \subseteq A^n$ as follows:

- if ϕ equals \top then $\phi^{\underline{A}} = A^n$;
- if ϕ equals $t_1 = t_2$ then $\phi^{\underline{A}}$ is the relation

$$\{(a_1, \dots, a_n) \mid t_1^{\underline{A}}(a_1, \dots, a_n) = t_2^{\underline{A}}(a_1, \dots, a_n)\};$$
- if ϕ equals $R(t_1, \dots, t_k)$ then

$$\phi^{\underline{A}} := \{(a_1, \dots, a_n) \mid (t_1^{\underline{A}}(\bar{a}), \dots, t_k^{\underline{A}}(\bar{a})) \in R^{\underline{A}}\};$$
- if ϕ equals $\phi_1 \wedge \phi_2$ then $\phi^{\underline{A}} := \phi_1^{\underline{A}} \cap \phi_2^{\underline{A}}$;
- if ϕ equals $\neg \psi$ then $\phi^{\underline{A}} := A^n \setminus \psi^{\underline{A}}$;
- if ϕ equals $\exists x. \psi(x, x_1, \dots, x_n)$ then

$$\phi^{\underline{A}} := \{(a_1, \dots, a_n) \mid \text{there exists } a \in A \text{ such that } (a, a_1, \dots, a_n) \in \psi^{\underline{A}}\}.$$

A relation $R \subseteq A^k$ is called (*first-order*) *definable (in \underline{A})* if there exists a τ -formula $\phi(x_1, \dots, x_k)$ such that $R = \phi^{\underline{A}}$; we also say that ϕ *defines* R over \underline{A} . We say that two τ -formulas $\phi(x_1, \dots, x_n)$ and $\psi(x_1, \dots, x_n)$ are *equivalent* if for every τ -structure \underline{A} we have $\phi^{\underline{A}} = \psi^{\underline{A}}$. For $\phi(x_1, \dots, x_n)$ we write

$$\underline{A} \models \phi(a_1, \dots, a_n)$$

instead of $(a_1, \dots, a_n) \in \phi^{\underline{A}}$. In particular, if ϕ is a sentence, i.e., if $n = 0$, we write $\underline{A} \models \phi$, and say that \underline{A} *satisfies* ϕ , if $() \in \phi^{\underline{A}}$ (that is, if $\phi^{\underline{A}} \neq \emptyset$). Otherwise, we write $\underline{A} \not\models \phi$. Clearly, $\underline{A} \not\models \phi$ if and only if $\underline{A} \models \neg \phi$.

EXAMPLE 14. The following statements about well-known structures follow straightforwardly from the definitions.

- $(\mathbb{Z}; <) \models 0 < 1$
- $(\mathbb{Q}; <) \models \forall x, y (x < y \Rightarrow \exists z (x < z \wedge z < y))$ (density)
- $(\mathbb{Z}; <) \not\models \forall x, y (x < y \Rightarrow \exists z (x < z \wedge z < y))$ \triangle

DEFINITION 3.2.3. Let \underline{A} be a τ -structure. An operation $f: A^k \rightarrow A$ is called (*first-order*) *definable* in \underline{A} if the relation $\{(a_0, a_1, \dots, a_n) \mid a_0 = f(a_1, \dots, a_n)\}$ is first-order definable in \underline{A} ; this relation is also called the *graph of f* .

Exercises.

- (23) Let $\phi(x_1, \dots, x_n)$ and $\psi(x_1, \dots, x_n)$ be τ -formulas. Show that ϕ and ψ are equivalent if and only if $\underline{A} \models \forall \bar{x} (\phi(\bar{x}) \Leftrightarrow \psi(\bar{x}))$ for all τ -structures \underline{A} .
- (24) Let \underline{G} be a group. Show that the constant operation $e^{\underline{G}}$ and the unary operation and $(^{-1})^{\underline{G}}$ are definable in $(G; \circ^{\underline{G}})$.
- (25) A formula is in *prenex normal form* if it is of the form $Q_1 x_1 \dots Q_n x_n. \phi$ where Q_i is either \exists or \forall and ϕ is without quantifiers. Show that every formula $\phi(y_1, \dots, y_n)$ is equivalent to a formula $\psi(y_1, \dots, y_n)$ in prenex normal form.

3.2.5. First-order theories. Let τ be a signature. A (τ -)theory is a set of first-order τ -sentences. Let \underline{A} be a τ -structure and T a τ -theory. Then \underline{A} is a *model of T* , in symbols $\underline{A} \models T$, if $\underline{A} \models \phi$ for all $\phi \in T$. We write $T \models \phi$, and say that T implies ϕ , if $\underline{A} \models \phi$ for every model \underline{A} of T . If neither $T \models \phi$ nor $T \models \neg\phi$, then we say that ϕ is *independent from T* . If S and T are τ -theories then we write $S \models T$ if every model of S is also a model of T .

EXAMPLE 15. The theory T_{AGroup} of abelian groups is over the signature $\tau_{\text{AGroup}} = \{+, -, 0\}$ and contains the following axioms:

- | | |
|--|--------------------|
| $\forall x, y, z. (x + y) + z = x + (y + z)$ | (associativity) |
| $\forall x. 0 + x = x + 0 = x$ | (neutral element) |
| $\forall x. x + (-x) = 0$ | (inverse elements) |
| $\forall x, y. x + y = y + x$ | (abelian) |

A τ_{AGroup} -structure is called a *group* if it is a model of T_{AGroup} . \triangle

EXAMPLE 16. The theory T_{CRing} of commutative rings is over the signature τ_{Ring} , contains T_{AGroup} and the following additional axioms:

- | | |
|---------------------------------------|-----------------------|
| $\forall x, y, z. (xy)z = x(yz)$ | (associativity) |
| $\forall x. 1 \cdot x = x$ | (multiplicative unit) |
| $\forall x, y. xy = yx$ | (commutativity) |
| $\forall x, y, z. x(y + z) = xy + xz$ | (distributivity) |

A τ_{CRing} -structure is called a *ring* if it is a model of T_{CRing} . \triangle

EXAMPLE 17. The theory T_{Field} of fields is over the signature τ_{Ring} , contains T_{CRing} and the following additional axioms:

- $$\neg(0 = 1)$$
- $$\forall x (\neg(x = 0) \Rightarrow \exists y. xy = 1)$$

A τ_{Ring} -structure is called a *field* if it is a model of T_{Field} . \triangle

DEFINITION 3.2.4. A τ -theory T is called

- *satisfiable* if T has a model,
- *complete* if T is satisfiable and for every τ -sentence either $T \models \phi$ or $T \models \neg\phi$, and
- *incomplete* if T is not complete.

The (*first-order*) *theory of \underline{A}* is defined as the set of all first-order τ -sentences that are satisfied by \underline{A} . Note that $\text{Th}(\underline{A})$ is always a complete theory.

The following game is designed for verifying that two structures have the same theory. If \underline{A} and \underline{B} are τ -structures, $f: S \rightarrow B$ is a map for some $S \subseteq A$, and $\phi(x_1, \dots, x_n)$ is a τ -formula, then we say that f *preserves ϕ* if for all $a_1, \dots, a_n \in S$ we have that $\underline{A} \models \phi(a_1, \dots, a_n)$ implies that $\underline{B} \models \phi(f(a_1), \dots, f(a_n))$.

DEFINITION 3.2.5 (Ehrenfeucht-Fraïssé Game). Let \underline{A} and \underline{B} be two countable structures with the same signature τ and $k \in \mathbb{N}$. The game $\text{EF}_k(\underline{A}, \underline{B})$ is played by the two players \exists and \forall in k rounds. In the first round, Player \forall starts by choosing an element of one of the structures. Player \exists has to respond by choosing an element in the other structure, which completes the first round. In the second round, player \forall again chooses an element from either A or B , and \exists again has to answer by choosing an element of the other structure. Let $(a_1, b_1), \dots, (a_k, b_k) \in A \times B$ be the pairs of elements that are fixed in the k rounds of the game. Player \exists wins this game if the mapping from A to B given by $a_i \mapsto b_i$, for $i \in [k]$, preserves all quantifier-free τ -formulas; otherwise, player \forall wins.

A *strategy* for a player in the game is a set of ‘rules’ which tell the player exactly how to move, depending on what has happened earlier in the play. A *winning strategy* is a strategy for a player such that for every strategy of the opponent, if the player follows her strategy she wins.

EXAMPLE 18. Player \forall has a winning strategy for $\text{EF}_3((\mathbb{Q}; <), (\mathbb{Z}; <))$. For every $k \in \mathbb{N}$, Player \exists has a winning strategy for $\text{EF}_k((\mathbb{Q}; <), (\mathbb{R}; <))$. \triangle

LEMMA 3.2.6. *Let \underline{A} and \underline{B} be two structures with the same signature τ . If \exists has a winning strategy for $\text{EF}_k(\underline{A}, \underline{B})$ for every k , then $\text{Th}(\underline{A}) = \text{Th}(\underline{B})$.*

Lemma 3.2.6 is intuitive and elegant; however, we still need to formalise the concept of a (*winning*) *strategy* for this game, and formally working with these winning strategies can sometimes be a bit cumbersome. There is a characterisation of the existence of a winning strategy for player \exists which is much easier to work with formally. For this reason, we do not prove Lemma 3.2.6, but only work with the following definition.

DEFINITION 3.2.7 (Back-and-Forth Equivalence). Let \underline{A} and \underline{B} be τ -structures and $k \in \mathbb{N}$. A *k -round back-and-forth system* from \underline{A} to \underline{B} is a non-empty set I of isomorphisms between substructures of \underline{A} of size at most k and substructures of \underline{B} such that for every $f \in I$ with domain size at most $k - 1$

- for any $c \in A$ there is an extension of f in I with c in its domain.
- for any $d \in B$ there is an extension of f in I with d in its image.

If there exists a back-and-forth system from \underline{A} to \underline{B} then \underline{A} and \underline{B} are said to be *k -round back-and-forth equivalent*.

EXAMPLE 19. If there exists an isomorphism i between \underline{A} and \underline{B} , then the set of all restrictions of i to subsets of A of size k is a k -round back-and-forth system. \triangle

EXAMPLE 20. For every $k \in \mathbb{N}$, the set of all isomorphisms between substructures of $(\mathbb{Q}; <)$ of size at most k and substructures of $(\mathbb{R}; <)$ is a back-and-forth system from $(\mathbb{Q}; <)$ to $(\mathbb{R}; <)$. \triangle

LEMMA 3.2.8. Let \underline{A} and \underline{B} be two structures with the same signature τ . If for every $k \in \mathbb{N}$ the structures \underline{A} and \underline{B} are k -round back-and-forth equivalent, then $\text{Th}(\underline{A}) = \text{Th}(\underline{B})$.

PROOF. Let ϕ be a τ -sentence. Let k be the number of variables in ϕ . By assumption, there exists a k -round back-and-forth system I from \underline{A} to \underline{B} . We prove by induction over the syntactic structure of first-order formulas that for every τ -formula $\phi(x_1, \dots, x_n)$ and elements a_1, \dots, a_n such that $\underline{A} \models \phi(a_1, \dots, a_n)$, if $f \in I$ is defined on a_1, \dots, a_n then $\underline{B} \models \phi(f(a_1), \dots, f(a_n))$. This is clearly true if ϕ is atomic because f is an isomorphism. If ϕ is of the form $\phi_1 \wedge \phi_2$ then the statement follows from the inductive assumption. If ϕ is of the form $\neg\phi$ then the statement follows from the inductive assumption, because the statement is symmetric with respect to \underline{A} and \underline{B} . If ϕ is of the form $\exists x.\psi(x_1, \dots, x_n, x)$, then there exists $a \in A$ such that $\underline{A} \models \psi(a_1, \dots, a_n, a)$. Pick an extension f' of f from I which is also defined on a ; such an extension exists by the definition of k -round back-and-forth systems and since k is chosen large enough. By the inductive assumption, we have $\underline{B} \models \psi(f'(a_1), \dots, f'(a_n), f'(a))$ and hence $\underline{B} \models \phi(f(a_1), \dots, f(a_n))$, which concludes the induction. \square

Exercises.

- (26) Show that the theory of abelian groups is incomplete.
- (27) Show that $(\mathbb{Q}; +, -, 0, 1, \cdot)$ and $(\mathbb{R}; +, -, 0, 1, \cdot)$ do not have the same first-order theory.
- (28) Write down a first-order theory T such that there is a bijection between the models of T with domain B and the graphs with vertex set B .
- (29) Show that the sentence

$$\begin{aligned} & \forall x \exists y. P(x, y) \\ \wedge & \quad \forall x, y, z ((P(x, y) \wedge P(y, z)) \Rightarrow P(x, z)) \\ \wedge & \quad \forall x. \neg P(x, x) \end{aligned}$$

has no finite models (with at least one element), but infinite models.

- (30) Write down the axioms of algebraically closed fields in first-order logic.
- (31) Let \underline{A} and \underline{B} be τ -structures and let ϕ be a τ -formula which is *existential positive*, i.e., without universal quantifiers and without negation. Then ϕ is preserved by all homomorphisms from \underline{A} to \underline{B} .
- (32) Show that for every finite structure \underline{A} with a finite signature τ there exists a τ -sentence ϕ such that all τ -structures that satisfy ϕ are isomorphic to \underline{A} .
- (33) Show that the previous exercise is false if τ is infinite.
- (34) Show that two finite structures (no restriction is made on the cardinality of the signature) are isomorphic if and only if they have the same theory.
- (35) Show that if a relation R has a first-order definition in a structure \underline{A} , then R is preserved by all *automorphisms* of \underline{A} . An automorphism of \underline{A} is an isomorphism between \underline{A} and \underline{A} .
- (36) Show that the following are equivalent.
 - (a) T is complete.
 - (b) T is *maximally satisfiable*, i.e., T is satisfiable, and for every first-order sentence ϕ , either $T \models \phi$ or $T \cup \{\phi\}$ is unsatisfiable.
 - (c) T is satisfiable, and $T = \text{Th}(\underline{A})$ for every $\underline{A} \models T$.
 - (d) $T = \text{Th}(\underline{A})$ for some structure \underline{A} ;
- (37) Show that the usual ordering $<$ of the real numbers is definable in the structure $(\mathbb{R}; +, \cdot, 0, 1)$.

- (38) Show that there is no linear order on the complex numbers which is definable in $(\mathbb{C}; +, \cdot, 0, 1)$.
- (39) Show that back-and-forth equivalence of structures is indeed an equivalence relation on structures.
- (40) Find a model of $\text{Th}(\mathbb{Z}; <)$ which is not isomorphic to $(\mathbb{Z}; <)$.
- (41) (*) Find a model of $\text{Th}(\mathbb{Z}; +)$ which is not isomorphic to $(\mathbb{Z}; +)$.
- (42) Show that if τ is a finite relational signature, then the converse implication in Lemma 3.2.8 is true as well.
- (43) Show that if we modify the notion of back-and-forth equivalence (Definition 3.2.7) by dropping the parameter k , i.e., we consider systems I of isomorphisms between arbitrary substructures of \underline{A} and of \underline{B} , then the existence of a back-and-forth system between two countable structures \underline{A} and \underline{B} implies that \underline{A} and \underline{B} are isomorphic.

Set Theory

This chapter is a very short introduction to Zermelo–Fraenkel set theory. See [17] for a more detailed introduction to set theory. We use ZF to denote the axioms of Zermelo and Fraenkel, which is a first-order theory (Section 4.1). The acronym ZFC stands for the extension of ZF by the axiom of choice; ZFC suffices for practically all of mathematics (ironically, set theory itself is the major field that studies statements that do not follow from ZFC). In particular, we can formalise the concepts of *ordinal* (Section 4.2) and *cardinal* (Section 4.3) in ZFC. The system ZF builds on Zermelo set theory (Z) which we introduce first.

4.1. ZFC

We work with the signature $\tau = \{\in\}$ where \in is a binary relation symbol that is used in infix notation. The elements $a \in U$ of a τ -structure \underline{U} are called *sets*, and ‘ $a \in b$ ’ is intended to mean ‘ a is an element of b ’. The domain U of \underline{U} is called a *universe of sets*; it is a set in the naive sense (on the meta level), and not a set in the sense above. We use the following abbreviations for τ -formulas:

- $x \notin y$ for $\neg(x \in y)$
- $x \subseteq y$ for $\forall z (z \in x \Rightarrow z \in y)$.
- $x \subset y$ for $x \subseteq y \wedge x \neq y$.

The Zermelo–Fraenkel axioms (ZF) is the following set of τ -sentences.

- (1) *Extensionality*. Sets containing the same elements are equal:

$$\forall x, y ((x \subseteq y \wedge y \subseteq x) \Rightarrow x = y)$$

- (2) *Empty set*. There exists an empty set (denoted by \emptyset):

$$\exists x \forall y (y \notin x)$$

- (3) *Pairing*. For all sets a and b there is a set (denoted by $\{a, b\}$) which has exactly the elements a and b :

$$\forall a, b \exists c \forall x (x \in c \Leftrightarrow (x = a \vee x = b))$$

- (4) *Union*. For every set a there is a set (denoted by $\bigcup a$; we write $a \cup b$ for $\bigcup\{a, b\}$) that contains precisely the elements of the elements of a :

$$\forall a \exists b \forall x (x \in b \Leftrightarrow \exists y \in a. x \in y)$$

- (5) *Power set*. For every set a there is a set (denoted by $\mathcal{P}(a)$) that consists of all subsets of a :

$$\forall a \exists b \forall x (x \in b \Leftrightarrow x \subseteq a)$$

- (6) *Infinity*. There is an infinite set. One way to express this is to assert the existence of a set which contains the empty set and is closed under the *successor operation* $x \mapsto x^+ := x \cup \{x\}$:

$$\exists a (\emptyset \in a \wedge \forall x (x \in a \Rightarrow x \cup \{x\} \in a)).$$

- (7) *Comprehension*: If $\phi(x, y_1, \dots, y_m)$ is a first-order $\{\in\}$ -formula, for all sets b_0, b_1, \dots, b_m there exists a subset of b_0 , denoted by

$$\{x \in b_0 \mid \phi(x, b_1, \dots, b_m)\}$$

containing precisely those elements x of b_0 that satisfy $\phi(x, b_1, \dots, b_m)$. Written as a first-order formula, this is

$$\forall y_0, y_1, \dots, y_m \exists z \forall x (x \in z \Leftrightarrow (x \in y_0 \wedge \phi(x, y_1, \dots, y_m))).$$

Comprehension is in fact an infinite family of axioms (for every first-order formula ϕ we have one axiom); one therefore speaks of the *axiom scheme of comprehension*. The axioms presented so far are sometimes called *Zermelo's set theory*, abbreviated by Z .

REMARK 4.1.1. The infinity axiom jointly with comprehension implies that there is a smallest set with respect to inclusion that contains the empty set and is closed under the successor operation, which we will then denote by ω . Note that the property of a set to contain \emptyset and to be closed under the successor operation can be expressed by a first-order formula $\phi(x)$. If a is any set that contains \emptyset and is closed under the successor operation (which exists by the axiom of infinity) then $\{x \in a \mid \forall y \phi(y) \Rightarrow x \in y\}$ clearly satisfies ϕ , and is contained in any set that is closed under the successor operation and contains \emptyset . So $\omega = \{\emptyset, \emptyset^+, (\emptyset^+)^+, \dots\}$. We define $0 := \emptyset$, $1 := \{0\}$, $2 := \{0, 1\}$, and so on. Thus, $\omega = \{0, 1, 2, \dots\}$.

REMARK 4.1.2 (Russel's Antinomy). Comprehension implies that there is no set containing all sets. Indeed, suppose for contradiction that $\underline{U} \models \exists a \forall z. z \in a$. We may then apply comprehension with the formula $\phi(x)$ given by $x \notin x$ and obtain the existence of the set $b := \{x \in a \mid x \notin x\}$. Then $b \in b$ if and only if $b \notin b$, a contradiction. This contradiction is called *Russel's Antinomy*.

Fraenkel and independently Skolem [8, 28] pointed out that Z is too weak as an axiom system for the intuitive notion of sets that we have in mathematics. For example, in Z one cannot show the existence of the set

$$E := \{Z_0, Z_1, Z_2, \dots\} \tag{6}$$

where $Z_0 := \omega$ and $Z_{i+1} := \mathcal{P}(Z_i)$ for all $i \in \mathbb{N}$: One can prove that $\bigcup E$, equipped with the restriction of \in to that set, provides a model of Z which does not have the element E . (Bonus exercise: verify this.)

Before proceeding with the axioms of ZF, we mention that the axioms presented so far are not independent: Axiom (2) about the existence of an empty set follows from Comprehension: for any set a , the set $\{b \in a \mid \neg(b = b)\}$ does not have elements (and by Extensionality, it is unique with this property); note that we use here the assumption that structures have non-empty domains.

To formulate the remaining axioms of ZF we need the set-theoretic concept of a function, which we introduce now. First, an *ordered pair* (also called *Kuratowski pair*) of two sets a and b is the set

$$(a, b) := \{\{a\}, \{a, b\}\}.$$

Exercises.

- (44) Show that if $(a, b) = (a', b')$ if and only if $a = a'$ and $b = b'$.
 (45) Do we also get the property formulated in the previous exercise if we would define $(a, b) := \{a, \{a, b\}\}$. If yes: prove it. If not, why not?

(46) Derive from ZF that for any two sets a, b there exists a *Cartesian product* $a \times b$ of a and b :

$$a \times b := \{(x, y) \mid x \in a \wedge y \in b\}.$$

DEFINITION 4.1.3. A *(binary) relation* is a set of ordered pairs. A *function* is a relation f which satisfies

$$\forall x, y, z ((x, y) \in f \wedge (x, z) \in f \Rightarrow y = z).$$

We write $f(x) = y$ instead of $(x, y) \in f$. Then

$$\text{dom}(f) := \{x \in \bigcup \bigcup f \mid \exists y. f(x) = y\}$$

is called the *domain* of f . Likewise,

$$\text{im}(f) := \{y \in \bigcup \bigcup f \mid \exists x. f(x) = y\}$$

is called the *image* of f . If $\text{im}(f) \subseteq b$, we write $f: \text{dom}(f) \rightarrow b$.

Note that if $\underline{U} \models Z$ and $S \subseteq U$ then there may be no element of U that contains precisely the elements of S . So S is not a set in the proper sense, only in the naive sense (on the meta level). Likewise, a function f in the naive sense might not be a function in the sense of Definition 4.1.3. The idea of the following axiom scheme is that if a function f in the meta sense is definable in \underline{U} , then the image $\{f(a) \mid a \in d\}$ of an arbitrary set d under f is a set.

(8) *Replacement*: For every first-order $\{\in\}$ -formula $\phi(v_1, \dots, v_n, x, y)$ we have the axiom

$$\begin{aligned} \forall d, v_1, \dots, v_n (\forall x, y_1, y_2 (\phi(\bar{v}, x, y_1) \wedge \phi(\bar{v}, x, y_2) \Rightarrow y_1 = y_2) \\ \Rightarrow \exists u \forall y (y \in u \Leftrightarrow \exists x \in d. \phi(\bar{v}, x, y))). \end{aligned}$$

EXAMPLE 21. Replacement implies the existence of the set E from (6). To see this, let \underline{U} be a model of the axioms of set theory that we have presented so far. Let $G: U \rightarrow U$ be a function in the naive sense, defined by

- $G(\emptyset) := \omega$,
- $G(f) := \mathcal{P}(\bigcup \{f(y) \mid y \in \{0, 1, \dots, n\}\})$ if $f: \{0, 1, \dots, n\} \rightarrow U$, and
- $G(x) := \emptyset$ otherwise.

Claim. There is a $\{\in\}$ -formula that defines in \underline{U} a function $F: \omega \rightarrow U$ in the naive sense such that $F(n) = G(F|_{\{0, \dots, n-1\}})$ where $F|_{\{0, \dots, n-1\}}$ denotes the restriction of F to $\{0, \dots, n-1\}$. We first show that for every $n \in \omega$ there exists a function $f: \{0, \dots, n\} \rightarrow U$ satisfying

$$\forall i \in \{0, \dots, n-1\} (f(i) = G(f|_{\{1, \dots, i\}})). \quad (7)$$

This can be shown by induction on n : define $f_0 := \emptyset$ and define $f_{n+1} := f_n \cup \{(n, G(f_n))\}$. Now, $F(n) = x$ if and only if there exists a function f with domain $\{0, 1, \dots, n+1\}$ satisfying (7) and $f(n) = x$, and the claim follows. Note that

$$\begin{aligned} F(0) &= G(\emptyset) = \omega \\ F(1) &= G(F|_{\{0\}}) = \mathcal{P}(F(0)) = \mathcal{P}(\omega), \\ F(2) &= G(F|_{\{0,1\}}) = \mathcal{P}(F(1)) = \mathcal{P}(\mathcal{P}(\omega)), \text{ etc.} \end{aligned}$$

By replacement, the claim implies that

$$\{F(a) \mid a \in \omega\} = \{F(0), F(1), F(2), \dots\} = \{\omega, \mathcal{P}(\omega), \mathcal{P}(\mathcal{P}(\omega)), \dots\}$$

is a set. △

We want to exclude that sets can be their own element. The following axiom is even stronger:

(9) *Foundation.*

$$\forall x (x \neq \emptyset \Rightarrow \exists y \in x. \neg \exists z (z \in x \wedge z \in y))$$

Foundation also excludes infinite sequences (a_n) such that a_{i+1} is an element of a_i for all i . Mathematics can be practised without this axiom, and some authors do not include this axiom in ZF. But assuming Foundation simplifies some proofs of fundamental properties of ordinals. This concludes the list of axioms of ZF.

For ZFC we add the *Axiom of Choice*.

(10) *Choice:* For every set a there is a function $c: \mathcal{P}(a) \setminus \{\emptyset\} \rightarrow a$ (called a *choice function*) such that $c(b) \in b$ for all non-empty $b \in \mathcal{P}(a)$.

Exercises.

- (47) Let a and b be sets. Then $\{f \mid f: a \rightarrow b\}$ is a set.
- (48) Show that Foundation indeed rules out the existence of a set a such that $a \in a$.
- (49) Show that Replacement implies Comprehension.
- (50) (*) Show that Pairing is a consequence of Power set and Replacement.
- (51) Prove the claims that are made in Example 21 to prove that ZF implies the existence of the set $\{\omega, \mathcal{P}(\omega), \mathcal{P}(\mathcal{P}(\omega)), \dots\}$.

4.1.1. Sets and classes. In set theory we distinguish between sets and *classes*. This is motivated by the desire to speak e.g. about of the *class of all τ -structures* or the *class of all ordinals*. However, we know that these things cannot be sets (as we will see in the next section, if the class of all ordinals were a set, we could derive a contradiction).

Using first-order logic, we therefore want to give an axiomatic treatment of set theory that allows for the distinction between sets and (proper) classes. This can be done using ZF as follows. Let $\underline{U} = (U; \in)$ be a model of ZF. We now consider the expansion \underline{U}_U of $(U; \in)$ (Definition 3.1.1). If $\phi(x)$ is a first-order $\{\in\} \cup \{c_u \mid u \in U\}$ -formula and $S := \{u \in U \mid \underline{U}_U \models \phi(u)\}$, then S is called a *class*. Note that any set $a \in U$ gives rise to a class C_a , namely the class given by the formula ' $x \in c_a$ '. Note that by Extensionality, $a = b$ if and only if $C_a = C_b$. Classes that are not sets are called *proper classes*.

REMARK 4.1.4. There are other axiom systems for set theory, for example *Bernays–Gödel set theory (BG)*. The difference concerns mainly how to implement the distinction between sets and classes. This distinction is more explicit in Bernays–Gödel set theory, which is formulated using ‘two-sorted structures’. One sort of objects are *sets* and the other sort of objects are *proper classes*; sets can be elements of sets, and sets can be elements of proper classes, but proper classes cannot be elements of sets or proper classes.

4.1.2. Partial orders and Zorn’s lemma. Let A be a set. A (*binary*) *relation over A* is a subset of $A^2 = A \times A$. A binary relation \leq over A is called a *partial order (on A)* if it is

- *reflexive:* $\forall x \in A. x \leq x$.
- *antisymmetric:* $\forall x, y \in A ((x \leq y \wedge y \leq x) \Rightarrow x = y)$.
- *transitive:* $\forall x, y, z \in A ((x \leq y \wedge y \leq z) \Rightarrow x \leq z)$.

We write $x < y$ as a shortcut for $x \leq y \wedge x \neq y$. We sometimes refer to $<$ as a partial order, too; in this case, we mean that the relation \leq defined from $<$ by $x \leq y$ iff $x < y$ or $x = y$ is a partial order.

A *chain* in a partial order \leq on A is subset of A such that for all $x, y \in S$ we have $x \leq y$ or $y \leq x$. If all of A is a chain, then the partial order \leq on A is called a *weak linear order*, and the relation $<$ on A is then called a *strict linear order*. Sometimes the addition of ‘*weak*’ and ‘*strict*’ is omitted, but since it is clear how to pass from weak to strict linear orders and vice versa, this should not cause problems. An element $a \in A$ is called an *upper bound* of $S \subseteq A$ if $s \leq a$ for all $s \in S$. An element $a \in A$ is called a *largest element* if it is an upper bound for all subsets of A and it is called *maximal* if there is no element $b \in A$ such that $a < b$. Note that a partial order can have at most one largest elements, but may have many maximal elements. *Lower bounds*, *smallest elements*, and *minimal elements* are defined analogously.

The following example shows how new partial orders can be constructed from known partial orders; this will be needed later in the section on ordinals to define ordinal addition, multiplication, and exponentiation (Section 4.2).

EXAMPLE 22. Let $<_1$ be a partial order on X_1 and $<_2$ be a partial order on X_2 .

- The *ordered sum* of $<_1$ and $<_2$ is the partial order $<$ defined on the set $(X_1 \times \{1\}) \cup (X_2 \times \{2\})$ by $(a, i) < (b, j)$ if $i < j$ or if $i = j$ and $a <_i b$.
- The *reverse lexicographic product* of $<_1$ and $<_2$ is defined as the partial order $<$ on $X_1 \times X_2$ defined as follows: $(x, y) < (x', y')$ if $y <_2 y'$ or if $y = y'$ and $x <_1 x'$.
- If X_1 has a smallest element 0. Then the set $X_1^{(X_2)}$ of functions from X_2 to X_1 with finite *support* $\{x \in X_2 \mid f(x) \neq 0\}$ can be partially ordered by setting $f < g$ if there exists $x \in X_2$ such that $f(x) <_1 g(x)$ and $f(x') = g(x')$ for every $x' \in X_1$ with $x <_2 x'$. \triangle

In ZFC one can prove Zorn’s lemma, which will be used later in this text. We defer the proof to the next section when we have the concept of ordinals available.

THEOREM 4.1.5 (Zorn’s Lemma). *Let $(P; \leq)$ be a partially ordered set with the property that every chain in P has an upper bound in P . Then P contains at least one maximal element.*

4.2. Ordinals

We freely follow Hils and Loeser [13]. Ordinals are a natural extension of the natural numbers. They classify well-ordered sets and are a natural device for using transfinite induction.

4.2.1. Well-orders. A partial order \leq on a set X is *well-founded* if any non-empty subset of X contains a smallest element with respect to \leq . A *well-order* on a set X is a strict linear order $<$ on X such that the relation \leq , defined as usual by $x \leq y$ iff $x < y$ or $x = y$, is well-founded. The usual ordering $<$ of the set \mathbb{N} is an example of a well-order, and the usual ordering $<$ of \mathbb{Z} and the usual ordering $<$ of the non-negative rational numbers are non-examples.

LEMMA 4.2.1. *If X and Y are linearly ordered sets, then the ordered sum and the reverse lexicographic product of X and Y and the partial order defined on $X^{(Y)}$ in Example 22 are linear orders as well. If X and Y are well-founded partial orders, then the ordered sum, the reverse lexicographic product, and the partial order on $X^{(Y)}$ are well-founded as well.*

PROOF. The only non-trivial points to check are the well-foundedness of the reverse lexicographic product and of $X^{(Y)}$. Let Z be a non-empty subset of $X \times Y$. Since Y is well-founded, the set $\{y \mid (x, y) \in Z\}$ has a minimal element y' . Since X is well-founded, the set $\{x \in X \mid (x, y) \in Z\}$ contains a minimal element x' . Then (x', y') is minimal in Z .

Let Z be a non-empty subset of $X^{(Y)}$. If the constant function with value 0 belongs to Z , then this function is a minimal element in Z . So we may assume that every $f \in Z$ has a non-empty support $\text{Supp}(f)$. Let

$$\begin{aligned} Y_1 &:= \{\max(\text{Supp}(f)) \mid f \in Z\} \\ y_1 &:= \min(Y_1) \\ Z'_1 &:= \{f \in Z \mid y_1 = \max(\text{Supp}(f))\} \\ x_1 &:= \min(\{f(y_1) \mid f \in Z'_1\}) \\ Z_1 &:= \{f \in Z'_1 \mid f(y_1) = x_1\} \end{aligned}$$

Now suppose inductively that for $n \geq 1$ we have already constructed $Y_1, \dots, Y_n, y_1, \dots, y_n, Z'_1, \dots, Z'_n, x_1, \dots, x_n$, and Z_1, \dots, Z_n . If Z_n contains the function with constant value 0 outside $\{y_1, \dots, y_n\}$, then we have found our minimal element in Z . Otherwise, we have $\text{Supp}(f) \setminus \{y_1, \dots, y_n\} \neq \emptyset$ for every $f \in Z_n$. Let

$$\begin{aligned} Y_{n+1} &:= \{\max(\text{Supp}(f) \setminus \{y_1, \dots, y_n\}) \mid f \in Z_n\}, \\ y_{n+1} &:= \min(Y_{n+1}), \\ Z'_{n+1} &:= \{f \in Z_n \mid y_{n+1} = \max(\text{Supp}(f) \setminus \{y_1, \dots, y_n\})\} \\ x_{n+1} &:= \min(\{f(y_{n+1}) \mid f \in Z'_{n+1}\}) \\ Z_{n+1} &:= \{f \in Z'_{n+1} \mid f(y_{n+1}) = x_{n+1}\}. \end{aligned}$$

If Z_{n+1} contains the function with constant value 0 outside $\{y_1, \dots, y_n\}$ then we have found our minimal element in Z . Since the sequence (y_i) is strictly decreasing in Y , this case must eventually arise for some finite $n \in \mathbb{N}$. \square

4.2.2. Definition of ordinals.

DEFINITION 4.2.2 (von Neumann). A set a is

- *transitive* if for all sets b and c , if $c \in b$ and $b \in a$ then $c \in a$.
- an *ordinal* if it is transitive and if the relation $\{(x, y) \in a \times a \mid x \in y\}$ on a defines a well-order on a .

Note that a is transitive if and only if it satisfies $\forall b (b \in a \Rightarrow b \subseteq a)$. The property that $\{(x, y) \in a \times a \mid x \in y\}$ is a well-order on a can be expressed in first-order logic as well; note that well-foundedness can be written as

$$\forall y \in \mathcal{P}(a) (y \neq \emptyset \Rightarrow \exists z \in y. \neg \exists x (x \in z \wedge x \in y))$$

Hence, there is a $\{\in\}$ -formula $\text{Ord}(x)$ which holds for $c \in U$, for some model $(U; \in)$ of ZF, if and only if c is an ordinal; we may thus speak about the *class of all ordinals*.

From now on, we typically use α, β, γ , etc., to denote ordinals. An element β of an ordinal α is again an ordinal number: Since $\beta \subseteq \alpha$, we obtain a well-order of β by restricting the well-order of α to β .

PROPOSITION 4.2.3. *Let α and β be ordinals. Then $\alpha \in \beta$ if and only if $\alpha \subset \beta$.*

PROOF. Clearly, if $\alpha \in \beta$ then $\alpha \subseteq \beta$ by transitivity, and $\alpha \neq \beta$ because $\alpha \notin \alpha$. Conversely, if $\alpha \subset \beta$, then let γ be the smallest element of $\beta \setminus \alpha$ which exists because β is well-ordered.

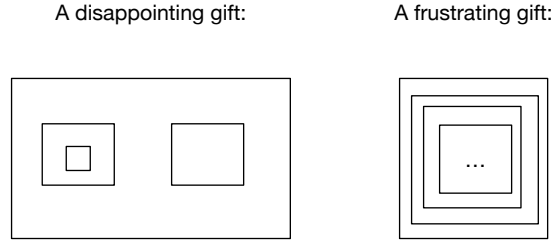


FIGURE 4.1. Ordinals are disappointing, but not frustrating gifts (Moshovakis [22]).

Claim 1. $\gamma \subseteq \alpha$: let $\delta \in \gamma$. Since $\gamma \in \beta$, we have $\delta \in \beta$. The minimality of γ implies that $\delta \notin \beta \setminus \alpha$, hence, $\delta \in \alpha$.

Claim 2. $\alpha \subseteq \gamma$. Let $\delta \in \alpha$. If $\gamma \in \delta$ we would have $\gamma \in \alpha$ contradicting the definition of γ . Since $\delta, \gamma \in \beta$ and β is totally ordered, this implies that $\delta \in \gamma$.

The claims imply that $\alpha = \gamma \in \beta$. □

PROPOSITION 4.2.4. *Let X be a non-empty set of ordinals. Then*

$$\bigcap X := \{\alpha \in \bigcup X \mid \forall \beta \in X: \alpha \in \beta\}$$

is an element of X which is smallest with respect to \in .

PROOF. Clearly, the intersection of transitive sets is transitive and the restriction of a well-order to a subset is a well-order. Hence, $\beta := \bigcap X$ is an ordinal. We have $\beta \subseteq \alpha$ for every $\alpha \in X$. If $\beta \notin X$, then $\beta \in \alpha$ for every $\alpha \in X$ by Proposition 4.2.3. It follows that $\beta \in \beta$, a contradiction. □

PROPOSITION 4.2.5. *Let α and β be ordinals. Then exactly one of the following holds:*

$$\alpha \in \beta, \quad \alpha = \beta, \quad \beta \in \alpha.$$

PROOF. It is clear that the three cases are mutually exclusive. We apply Proposition 4.2.4 to $X := \{\alpha, \beta\}$ and obtain that $\alpha \cap \beta = \alpha$ or $\alpha \cap \beta = \beta$. In the former case, we have $\alpha \subseteq \beta$ and hence $\alpha = \beta$ or $\alpha \in \beta$ by Proposition 4.2.3. In the latter case we conclude analogously that $\alpha = \beta$ or $\beta \in \alpha$. □

So the class of all ordinals is linearly ordered by \in . We have to be careful, though.

PROPOSITION 4.2.6 (Burali-Forti paradox). *There is no set that contains precisely the ordinal numbers.*

PROOF. If there were such a set α , then it would be an ordinal itself, and thus $\alpha \in \alpha$, contradicting the assumption that \in is a well-order on the elements of α . □

If α and β are ordinals, then we also write

- $\alpha < \beta$ if $\alpha \in \beta$ and
- $\alpha \leq \beta$ if $\alpha \subseteq \beta$.

This notation is often more suggestive because it emphasises transitivity of \in for ordinals. Moreover, if $(\alpha_i)_{i \in I}$ is a sequence of ordinals for some index set I , we write $\sup_{i \in I} \alpha_i$ for $\bigcup \{\alpha_i \mid i \in I\}$. A function $f: P \rightarrow Q$ between two ordered sets P and Q is called *strictly increasing* if $a, b \in P$ and $a < b$ implies $f(a) < f(b)$.

LEMMA 4.2.7. *Let $f: \alpha \rightarrow \beta$ be a map between two ordinals which is strictly increasing. Then $\gamma \leq f(\gamma)$ for every $\gamma \in \alpha$.*

PROOF. Suppose for contradiction that there is $\gamma \in \alpha$ such that $f(\gamma) < \gamma$; choose γ to be minimal. Since f is strictly increasing, $f(f(\gamma)) < f(\gamma)$. This contradicts the minimality of γ , because $\beta := f(\gamma) \in \alpha$ also satisfies $f(\beta) = f(f(\gamma)) < f(\gamma) = \beta$, but $\beta = f(\gamma) < \gamma$. \square

PROPOSITION 4.2.8. *Every structure $(A; <)$ where A is well-ordered by $<$ is isomorphic to $(\alpha; <)$ for some ordinal α . Moreover, the ordinal α and the isomorphism are unique.*

PROOF. Define $f: A \rightarrow \alpha$ inductively by $f(y) := \{f(z) \mid z < y\}$. The image of f is an ordinal α such that $(\alpha; <)$ is isomorphic to $(A; <)$ via f .

Uniqueness: suppose that $f': A \rightarrow \alpha'$ is an isomorphism between $(A; <)$ and $(\alpha'; <)$ for some ordinal α' . Then $g := f' \circ f^{-1}: \alpha \rightarrow \alpha'$ is an isomorphism as well. Lemma 4.2.7 implies that $g(\beta) \leq \beta$ for every $\beta \in \alpha$. We may argue analogously for g^{-1} and obtain that $g(\beta) = \beta$ and $\alpha = \alpha'$. \square

4.2.3. Successor and limit ordinals. Recall the definition of the successor function from the infinity axiom. Note that for any ordinal α the successor $\alpha^+ := \alpha \cup \{\alpha\}$ of α is the smallest ordinal greater than α . Starting from the smallest ordinal $0 := \emptyset$, its successor is $1 := \{0\}$, then $2 := \{0, 1\}$, and so on, yielding the natural numbers \mathbb{N} . When we view \mathbb{N} as an ordinal, we denote it by ω (see Remark 4.1.1). The next ordinal is $\omega^+ := \{0, 1, \dots, \omega\}$, etc. By definition, a successor ordinal β contains a maximal element α (so $\beta = \alpha^+$). Ordinals greater than 0 which are not successor ordinals are called *limit ordinals*. We let $\text{Lim}(x)$ be a formula which defines the class of all limit ordinals.

PROPOSITION 4.2.9. *A non-empty ordinal λ is a limit ordinal if and only if*

$$\lambda = \sup_{\alpha < \lambda} \alpha.$$

PROOF. If λ is a limit ordinal, let $\beta := \sup_{\alpha < \lambda} \alpha$. Clearly, $\beta \subseteq \lambda$. To show that $\lambda \subseteq \beta$, let $\alpha \in \lambda$. Then $\alpha^+ \leq \lambda$ and it follows that $\alpha^+ < \lambda$ because λ is not a successor ordinal. Thus, $\alpha^+ \subseteq \beta$ by the definition of β , and $\alpha \in \alpha^+ \subseteq \beta$.

If $\lambda = \sup_{\alpha < \lambda} \alpha$, then $\sup_{\alpha < \lambda} \alpha = \sup_{\alpha < \gamma} \alpha = \gamma < \lambda$. \square

Any ordinal can be written uniquely as $\lambda \underbrace{+\dots+}_{n \text{ times}}$ where λ is a limit ordinal or 0.

PROPOSITION 4.2.10 (Transfinite induction). *Let \underline{U} be a model of ZF, and let $\phi(x)$ be a $\{\in\} \cup U$ -formula. Then \underline{U}_U satisfies the following induction property:*

$$(\phi(0) \wedge \forall \gamma (\phi(\gamma) \Rightarrow \phi(\gamma^+)) \wedge \forall \gamma ((\text{Lim}(\gamma) \wedge \forall \delta \in \gamma. \phi(\delta)) \Rightarrow \phi(\gamma))) \Rightarrow \forall \gamma. \phi(\gamma)$$

PROOF. Suppose that $\underline{U}_U \models \neg \phi(\alpha)$ for some ordinal α . Choose α to be minimal with this property. Then either $\alpha = 0$, or α is a successor ordinal, or α is a limit ordinal, and in each of the cases one of the preconditions of the implication in the statement does not hold. \square

Exercises.

- (52) Show that if α, β are ordinals such that $\alpha < \beta$, then $\alpha^+ \subseteq \beta$.
- (53) Show that if α, β are ordinals, and $\emptyset < \alpha < \beta$, then $\beta \setminus \alpha$ is not an ordinal.
- (54) Let α and β be ordinals. Then $\alpha \cup \beta$ and $\bigcup(\beta \setminus \alpha)$ are ordinals.
- (55) Let α be an ordinal. Show that
 - $\bigcup \alpha \subseteq \alpha$.
 - $\bigcup \alpha$ is an ordinal.
- (56) If X is a set of ordinals, then $\bigcup X$ is an ordinal.

4.2.4. Ordinal arithmetic. If α and β are ordinals, then the ordered sum of α and β is a well-order by Lemma 4.2.1, and by Proposition 4.2.8 it is isomorphic to a unique ordinal, which is denoted by $\alpha + \beta$. Similarly, $\alpha\beta$ is defined as the unique ordinal isomorphic to the reverse lexicographic product of α and β , and α^β as the unique ordinal isomorphic to $\alpha^{(\beta)}$.

PROPOSITION 4.2.11 (Ordinal addition). *Let α, β be ordinals. Then $\alpha + 0 = \alpha$ and $\alpha + \beta^+ = (\alpha + \beta)^+$. If λ is a limit ordinal, then $\alpha + \lambda = \sup_{\beta < \lambda} (\alpha + \beta)$.*

PROOF. Clearly, $\alpha + 0 = \alpha$ and $\alpha + 1 = \alpha^+$. Then $\alpha + \beta^+ = (\alpha + \beta)^+$ follows from the easy fact that $\alpha + (\beta + \gamma) = (\alpha + \beta) + \gamma$. Now let λ be a limit ordinal. We first show that $\alpha + \lambda \geq \sup_{\beta < \lambda} (\alpha + \beta)$. This is immediate from the fact that $\alpha + \lambda \geq \alpha + \beta$ for every $\beta < \lambda$. The fact can be seen as follows: $\lambda \setminus \beta$ is a well-ordered set, and hence isomorphic to an ordinal δ by Proposition 4.2.8. Thus, $\alpha + \lambda = \alpha + \beta + \delta > \alpha + \beta$.

We finally show that $\alpha + \lambda \leq \sup_{\beta < \lambda} (\alpha + \beta)$. Let $\mu \in (\alpha + \lambda)$, i.e., $\alpha \leq \mu < \alpha + \lambda$. The $\mu = \alpha + \delta$ for some δ with $0 \leq \delta < \lambda$. Since λ is a limit ordinal, one has $\delta^+ < \lambda$, hence $\mu < \alpha + \delta^+ \leq \sup_{\beta < \lambda} (\alpha + \beta)$. \square

Note that by transfinite induction on β , the statements in Proposition 4.2.11 characterise $\alpha + \beta$ uniquely. One can show by induction that $1 + \alpha = \alpha + 1$ if α is finite. Otherwise, if α is infinite, then $1 + \alpha = \alpha$. The reason is that

$$1 + \omega = 1 + \sup_{\beta < \omega} \beta = \sup_{\beta < \omega} (1 + \beta) = \sup_{\beta < \omega} \beta^+ = \omega.$$

PROPOSITION 4.2.12 (Ordinal multiplication). *Let α, β be ordinals. Then $\alpha 0 = 0$ and $\alpha\beta^+ = \alpha\beta + \alpha$. If λ is a limit ordinal, then $\alpha\lambda = \sup_{\beta < \lambda} (\alpha\beta)$.*

PROOF. The first statement is easy. Let λ be a limit ordinal. If $\alpha = 0$ the statement is clear since $0\lambda = 0$. So we assume that $\alpha \neq 0$. The inequality $\alpha\lambda \geq \sup_{\beta < \lambda} \alpha\beta$ can be shown similarly as in the proof of Proposition 4.2.11. Conversely, let $\gamma < \alpha\lambda$. We claim that then there are unique ordinals ρ, μ with $\rho < \mu$ such that $\gamma = \alpha\mu + \rho$. This is referred to as *Euclidean division*. If $\gamma = 0$ there is nothing to prove. Otherwise, consider the mapping $f_0: \gamma \rightarrow \alpha \times \beta$ defined by $x \mapsto (0, x)$, which is strictly increasing. Hence, $\gamma \leq \alpha\gamma$ by Lemma 4.2.7. If $\gamma = \alpha\gamma$ one sets $\mu = \gamma$ and $\rho = 0$. Otherwise, $\gamma \in \alpha\gamma$. Let f be the unique isomorphism of ordered sets between $\alpha\gamma$ and $\alpha \times \gamma$. One sets $(\rho, \mu) = f(\beta)$. Since $\{(\rho', \mu') \mid \alpha < (\rho', \mu') < (\rho, \mu)\}$ is isomorphic to $(\alpha \times \mu) + \rho$ it follows that $\gamma = \alpha\mu + \rho$, finishing the proof of the claim.

Since $\mu < \lambda$, we have $\mu^+ < \lambda$ because λ is a limit ordinal, hence

$$\gamma = \alpha\mu + \rho < \alpha\mu + \alpha = \alpha\mu^+ \leq \sup_{\beta < \lambda} \alpha\beta. \quad \square$$

Again, transfinite induction on β shows that the statements in Proposition 4.2.12 characterise $\alpha\beta$ uniquely. Note that $2\omega = \omega < \omega + \omega = \omega 2$, because

$$2\omega = 2 \sup_{\beta < \omega} \beta = \sup_{\beta < \omega} 2\beta = \sup_{\beta < \omega} \beta.$$

PROPOSITION 4.2.13 (Ordinal exponentiation). *Let α, β be ordinals. Then $\alpha^0 = 1$ and $\alpha^{\beta^+} = \alpha^\beta \alpha$. If λ is a limit ordinal, then $\alpha^\lambda = \sup_{\beta < \lambda} \alpha^\beta$.*

PROOF. The first statement can be checked directly. Let λ be a limit ordinal. Then inequality $\alpha^\lambda \geq \sup_{\beta \in \lambda} \alpha^\beta$ is easy to show. For the converse inequality, let $f \in \alpha^{(\lambda)}$. One may assume that f is not the constant function with value 0. Then $\max(\text{Supp}(f)) < \lambda$, and hence $\beta := \max(\text{Supp}(f))^+ < \lambda$, which proves that there exists a strictly increasing function $\{g \in \alpha^{(\lambda)} \mid g \leq f\} \rightarrow \alpha^{(\beta)}$; hence, $f < \bigcup_{\beta \in \lambda} \alpha^\beta$ by Lemma 4.2.7. \square

Transfinite induction on β shows that the statements in Proposition 4.2.13 characterise α^β uniquely.

4.2.5. The well-ordering theorem.

THEOREM 4.2.14 (Well-ordering theorem). *Every set has a well-ordering.*

PROOF. Let A be a set. Fix a set B which does not belong to A and define a function f from the class of all ordinals to $A \cup \{B\}$ as follows:

- if α is an ordinal such that $A \setminus \{f(\beta) \mid \beta < \alpha\} \neq \emptyset$ then set $f(\alpha)$ to be an element from this set (here we use the Axiom of Choice).
- Otherwise, $f(\alpha) := B$.

Then $\gamma := \{\alpha \mid f(\alpha) \neq B\}$ is an ordinal and the restriction of f to γ is bijection between γ and A . \square

Note that the ordinal γ in the construction of the well-order of A is not unique, unless A is finite. The well-ordering theorem is in fact equivalent to the Axiom of Choice.

PROPOSITION 4.2.15. *ZF and the well-ordering theorem imply the Axiom of Choice.*

PROOF. Let A be a set. By hypothesis, A may be well-ordered, say by $<$. The function $f: \mathcal{P}(A) \setminus \{\emptyset\} \rightarrow A$ which associates to any non-empty subset of A its smallest element is a choice function on A . \square

PROOF OF ZORN'S LEMMA (THEOREM 4.1.5). We define inductively a weakly increasing sequence (x_α) in P , indexed by the ordinals. Let x_0 be any element of P . Now suppose that α is an ordinal such that x_β has already been defined for $\beta < \alpha$. If $\alpha = \beta + 1$ then let x_α be x_β if x_β is maximal, and some element of P which is larger than x_β otherwise; such an element exists by the axiom of choice. Otherwise, we take x_α to be an upper bound of the chain $(x_\beta)_{\beta < \alpha}$, which exists by assumption. Note that if P has no maximal elements then the sequence x_α is strictly increasing. But it cannot be strictly increasing, because then the ordinals would be in bijection with a subset of P , contradicting the fact that one cannot form the set of all ordinals (Proposition 4.2.6). This proves that P has a maximal element and completes the proof of Zorn's lemma. \square

4.3. Cardinals

Cardinals are used to compare the 'size' of sets, building on the notion of ordinals. We may use the existence of an injective, surjective, or bijective function between two sets to compare their size. The famous results of Cantor-Bernstein (Theorem 4.3.1) and Cantor (Theorem 4.3.2) illustrate this idea.

In the proof of the next theorem we use the following notation for functions $g: B \rightarrow B$: we define $g^0 := \text{id}$ for $i \in \mathbb{N}$ we inductively define $g^{i+1} := g^{i+1} \circ g$ where \circ denotes function composition.

THEOREM 4.3.1 (Cantor-Bernstein). *Let A and B be sets and let $f: A \rightarrow B$ and $g: B \rightarrow A$ be injective maps. Then there exists a bijection $h: B \rightarrow A$.*

PROOF. We may assume that $A \subseteq B$ and that f is the inclusion map. Now set $C := \{g^n(x) \mid n \in \mathbb{N}, x \in B \setminus A\}$. Note that $B \setminus C \subseteq A$ because $g^0(B \setminus A) = B \setminus A$. Define $h: B \rightarrow A$ by $h(x) := g(x) \in C$ if $x \in C$ and $h(x) := x$ if $x \in B \setminus C$. The map h is clearly injective and also surjective: indeed, any $y \in A \cap C$ is of the form $y = g(x)$ for some $x \in C$, and $x = h(x)$ for any $x \in A \setminus C$. \square

We say that two sets A and B *have the same cardinality*, and write $A \sim B$, if there exists a bijection between them. We say that A *has at most the cardinality of B* , and write $A \preceq B$, if there exists an injection from A to B . The theorem of Cantor-Bernstein states that if $A \preceq B$ and $B \preceq A$, then $A \sim B$.

EXAMPLE 23. The sets \mathbb{N}^2 and \mathbb{N} have the same cardinality. A bijection $\mathbb{N}^2 \rightarrow \mathbb{N}$ is given by $(x, y) \mapsto \frac{(x+y+1)(x+y)}{2} + x$. \triangle

THEOREM 4.3.2 (Cantor). *Let A be a set. Then there is no surjection $A \rightarrow \mathcal{P}(A)$.*

PROOF. Let $f: A \rightarrow \mathcal{P}(A)$ be a function. Consider the set

$$B := \{x \in A \mid x \notin f(x)\}.$$

For every $x \in A$ with $f(x) = B$ we have that $x \in B$ if and only if $x \notin B$; hence, B does not belong to the image of f . \square

By the well-ordering theorem, every set has the same cardinality as some ordinal. We call the smallest such ordinal *the cardinality of A* , denoted by $|A|$. Ordinals occurring in this way are called *cardinals*. An ordinal α is a cardinal if and only if all smaller ordinals do not have the same cardinality.

Notes.

- All natural numbers and ω are cardinals.
- $\omega + 1$ is the smallest ordinal that is not a cardinal.
- The cardinality of a finite set is a natural number.
- A set of cardinality ω is called *countably infinite*.

We write κ^+ for the smallest cardinal greater than κ , the *successor cardinal* of κ . To avoid confusion, from now on the ordinal successor of an ordinal α will be denoted by $\alpha + 1$. Positive cardinals which are not successor cardinals are called *limit cardinals*.

PROPOSITION 4.3.3. *Let X be a set of cardinals. Then $\lambda := \bigcup_{\kappa \in X} \kappa$ is a cardinal.*

PROOF. If there is a $\kappa_0 \in X$ such that $\kappa \leq \kappa_0$ for all $\kappa \in X$, then $\bigcup_{\kappa \in X} \kappa = \kappa_0$ and the statement is true. Otherwise, for every $\kappa \in X$ there is a $\kappa' \in X$ with $\kappa < \kappa'$. For each ordinal α with $\alpha < \lambda$ we have that $\alpha \in \lambda$ and hence $\alpha \in \kappa$ for some $\kappa \in X$. By the above, there is a $\kappa' \in X$ such that $|\alpha| \leq \kappa < \kappa' \leq |\lambda|$. Thus, every ordinal smaller than λ has smaller cardinality than λ , and λ is a cardinal. \square

DEFINITION 4.3.4. The \aleph -*hierarchy* assigns to any ordinal α a cardinal \aleph_α as follows:

$$\aleph_\alpha := \begin{cases} \omega & \text{if } \alpha = 0 \\ \aleph_\beta^+ & \text{if } \alpha = \beta + 1 \\ \bigcup_{\beta < \alpha} \aleph_\beta & \text{if } \alpha \text{ is a limit ordinal.} \end{cases}$$

PROPOSITION 4.3.5. *Every infinite cardinal is of the form \aleph_α for some ordinal α .*

PROOF. Let κ be an infinite cardinal. It is easy to show by transfinite induction that the function from $\kappa + 1$ to $\aleph_{\kappa+1}$ given by $\beta \mapsto \aleph_\beta$ is strictly increasing. Thus, $\aleph_\kappa \geq \kappa$ by Lemma 4.2.7 and $\aleph_{\kappa+1} > \kappa$. Let $\alpha \leq \kappa + 1$ be minimal with $\aleph_\alpha > \kappa$. Since $\kappa \geq \aleph_0$, we have $\alpha > 0$. If α were a limit ordinal, by definition $\kappa \in \bigcup_{\beta < \alpha} \aleph_\beta$, and hence $\kappa \in \aleph_\beta$ for some $\beta < \alpha$, which would contradict the minimality of α . Thus, $\alpha = \beta + 1$ for some ordinal β , and $\aleph_\beta \leq \kappa < \aleph_{\beta+1} = \aleph_\beta^+$. It follows that $\aleph_\beta = \kappa$. \square

We conclude that $\alpha \mapsto \aleph_\alpha$ is an isomorphism between the class of ordinals and the class of all infinite cardinals.

Sums, products, and powers of cardinals are defined as the cardinality of disjoint unions, Cartesian powers, and sets of functions:

$$\begin{aligned} |x| + |y| &:= |x \uplus y| \\ |x| \cdot |y| &:= |x \times y| \\ |x|^{|y|} &:= |x^y| \end{aligned}$$

and likewise for infinite sums and products:

$$\begin{aligned} \sum_{x \in I} |x| &:= \left| \biguplus_{x \in I} x \right| \\ \prod_{x \in I} |x| &:= \left| \prod_{x \in I} x \right|. \end{aligned}$$

Note that

$$(\kappa^\lambda)^\mu = \kappa^{\lambda \cdot \mu}.$$

By Theorem 4.3.2 we have

$$2^\kappa > \kappa.$$

In particular, there is no largest cardinal. Cantor's result also follows from König's theorem below for $\kappa_i := 1$ and $\lambda_i := 2$ for all $i \in I$.

THEOREM 4.3.6 (König's theorem). *Let $(\kappa_i)_{i \in I}$ and $(\lambda_i)_{i \in I}$ be sequences of cardinals. If $\kappa_i < \lambda_i$ for all $i \in I$, then*

$$\sum_{i \in I} \kappa_i < \prod_{i \in I} \lambda_i.$$

PROOF. We first show that $\sum_{i \in I} \kappa_i \leq \prod_{i \in I} \lambda_i$. Choose pairwise disjoint sets $(A_i)_{i \in I}$ and $(B_i)_{i \in I}$ such that $|A_i| = \kappa_i$, $|B_i| = \lambda_i$, and $A_i \subset B_i$ for all $i \in I$. We will construct an injection $f: \bigcup_{i \in I} A_i \rightarrow \prod_{i \in I} B_i$. Choose $d_i \in B_i \setminus A_i$ for each $i \in I$ (here we use the Axiom of Choice). For $x \in A := \bigcup_{i \in I} A_i$, define

$$f(x) := (a_i)_{i \in I} \text{ where } a_i := \begin{cases} x & \text{if } x \in A_i \\ d_i & \text{otherwise.} \end{cases}$$

To show the injectivity of f , let $x, y \in A$ be distinct. Let $i \in I$ be such that $x \in A_i$. If $y \in A_i$ then $f(x)_i = x \neq y = f(y)_i$. If $y \notin A_i$ then $f(x)_i = x \neq d_i = f(y)_i$ since $x \in A_i$ but $d_i \in B_i \setminus A_i$. So in both cases, $f(x) \neq f(y)$.

Suppose for contradiction that $\sum_{i \in I} \kappa_i = \prod_{i \in I} \lambda_i$. Then we can find sets $(X_i)_{i \in I}$ with $|X_i| = \kappa_i$ such that

$$B := \prod_{i \in I} B_i = \bigcup_{i \in I} X_i.$$

For each $i \in I$, define

$$Y_i := \{a_i \mid a \in X_i\}.$$

For every $i \in I$ there exists $b_i \in B_i \setminus Y_i$ because $|Y_i| \leq |X_i| = \kappa_i < \lambda_i = |B_i|$. Now define

$$b := (b_i)_{i \in I} \in \prod_{i \in I} B_i.$$

Let $j \in I$. Then $b_j \notin Y_j$ by the choice of b_j , and hence $b \notin X_j$ by the definition of Y_j . This shows that $b \notin \bigcup_{i \in I} X_i$, a contradiction. \square

THEOREM 4.3.7. *Let κ be an infinite cardinal. Then*

$$(1) \quad \kappa \cdot \kappa = \kappa.$$

- (2) $\kappa + \lambda = \max(\kappa, \lambda)$.
 (3) $\kappa^\kappa = 2^\kappa$.

PROOF. For ordinals $\alpha, \beta, \alpha', \beta'$, define $(\alpha, \beta) < (\alpha', \beta')$ iff

$$(\max(\alpha, \beta), \alpha, \beta) <_{\text{lex}} (\max(\alpha', \beta'), \alpha', \beta')$$

where lex is the lexicographical order on triples of ordinals. Since this is a well-order, there is a unique order-preserving bijection f between pairs of ordinals and ordinals by Proposition 4.2.8.

Claim. If κ is an infinite cardinal, then f maps $\kappa \times \kappa$ to κ , and hence $\kappa \cdot \kappa = \kappa$. The proof of the claim is by induction on κ . For $\alpha, \beta \in \kappa$ let $P_{\alpha, \beta}$ be the set of predecessors of (α, β) . Note that:

- $P_{\alpha, \beta}$ is contained in $\delta \times \delta$ with $\delta = \max(\alpha, \beta) + 1$.
- Since κ is infinite and $\alpha, \beta < \kappa$, the cardinality of δ is smaller than κ .
- By inductive assumption $|P_{\alpha, \beta}| \leq |\delta \times \delta| = |\delta| \cdot |\delta| \stackrel{\text{IA}}{=} |\delta| < \kappa$.

Hence, $f(\alpha, \beta) < \kappa$ since f is an order isomorphism and thus $f(\alpha, \beta) \in \kappa$.

Now (2) and (3) are simple consequences. Let $\mu := \max(\kappa, \lambda)$.

$$\begin{aligned} \mu &\leq \kappa + \lambda \leq \mu + \mu \leq 2 \cdot \mu \leq \mu \cdot \mu = \mu \\ 2^\kappa &\leq \kappa^\kappa \leq (2^\kappa)^\kappa = 2^{\kappa \cdot \kappa} = 2^\kappa \end{aligned}$$

□

REMARK 4.3.8. The *Continuum Hypothesis (CH)* states that $\aleph_1 = 2^{\aleph_0}$, that is: there is no cardinal lying strictly between ω and the cardinality $|\mathbb{R}|$ of the continuum. The *Generalised Continuum Hypothesis (GCH)* states that $\kappa^+ = 2^\kappa$ for all infinite cardinals κ . As with CH, the GCH is known to be independent of ZFC, that is, there are models of ZFC where GCH is true, and models of ZFC where GCH is false (assuming that ZFC is consistent; see [17]).

Exercises.

- (57) Prove the claim in Example 23 that the given function from \mathbb{N}^2 to \mathbb{N} is bijective.
 (58) (*) (Exercise 1.11.5 in Hils and Loeser [13]) An *Ulam matrix* is a family

$$(U_{\alpha, n})_{\alpha < \aleph_1, n < \omega}$$

of subsets of \aleph_1 such that

- $U_{\alpha, n} \cap U_{\beta, n} = \emptyset$ for every $n < \omega$ and all distinct $\alpha, \beta \in \aleph_1$.
- $\aleph_1 \setminus \bigcup_{n < \omega} U_{\alpha, n}$ is countable for every $\alpha < \aleph_1$.

For $\xi < \aleph_1$, let $f_\xi: \omega \rightarrow \aleph_1$ be such that $\xi \subseteq \text{im}(f_\xi)$. Define

$$U_{\alpha, n} := \{\xi < \aleph_1 \mid f_\xi(n) = \alpha\}.$$

Prove that $(U_{\alpha, n})$ is an Ulam matrix.

- (59) (*) (Exercise 1.11.5 in Hils and Loeser [13]) Let $\mu: \mathcal{P}(\aleph_1) \rightarrow [0, 1]$ be a σ -additive measure on \aleph_1 , that is,

$$\mu\left(\bigcup_{n < \omega} A_n\right) = \sum_{n < \omega} \mu(A_n)$$

for all families $(A_n)_{n < \omega}$ of pairwise disjoint subsets of \aleph_1 . Prove that if $\mu(\{\alpha\}) = 0$ for every $\alpha < \aleph_1$, then μ is identically zero.

Hints:

- show that there are at most countably many pairwise disjoint subsets of \aleph_1 with positive measure.
- use the previous exercise.

(60) (*) (Exercise 1.11.10 in Hils and Loeser [13]) The goal of this exercise is to prove that the continuum hypothesis holds for closed subsets of the real line (Cantor's theorem), that is, if $S \subseteq \mathbb{R}$ is closed, then $|S| \in \{\aleph_0, 2^{\aleph_0}\}$. Suppose that $|S| > \aleph_0$. Let C be the set of $x \in S$ such that there is an open $U \subseteq \mathbb{R}$ which contains x and satisfies $|U \cap S| \leq \aleph_0$.

- Show that C is countable.
- Show that $S' := S \setminus C$ is a closed subset of \mathbb{R} .
- Show that any non-empty open subset of S' is uncountable.
- Let I be an open interval such that $I \cap S' \neq \emptyset$. Prove that for any $\epsilon > 0$ there are open intervals I_0 and I_1 of length at most ϵ such that $I_i \cap S' \neq \emptyset$ and $\bar{I}_i \subseteq I$ for $i \in \{0, 1\}$ and such that $\bar{I}_0 \cap \bar{I}_1 = \emptyset$.
- Prove that there is an injection from 2^{\aleph_0} into S' .
- Prove Cantor's theorem.

The Completeness Theorem

In this section we formalise the notion of a proof. A *formal proof* consists of a sequence of sentences; each sentence in the sequence is either a *logical axiom* or derived from previous sentences by a deduction principle called *modus ponens*. The sequence is then viewed as a proof of the final sentence in the sequence. The key features of our notion of a formal proof are that

- (1) every sentence that has a formal proof is valid (*soundness*),
- (2) every valid sentence has a formal proof (*completeness*), and
- (3) we can write a computer program that decides for a given sentence in the sequence whether it is a logical axiom or whether it can be derived from previously derived axioms via modus ponens (*effectivity*).

It is easy to adapt the proof system so that the running time of the algorithm is even polynomial in the size of the proof (*efficiency*).

Our set of axioms is infinite. It will be chosen so that the proof of the completeness theorem is as simple as possible. We essentially follow Hils and Loeser [13].

5.1. Logical Axioms

All our logical axioms are sentences that are valid (so the logical axioms should not be confused with the axioms of set theory, which clearly do not hold in all structures). We start with axioms that describe properties of equality. It is clear that the following first-order sentences are valid.

$$\begin{aligned} \forall x. x = x & && \text{(reflexivity, E1)} \\ \forall x, y (x = y \Rightarrow y = x) & && \text{(symmetry, E2)} \\ \forall x, y, z ((x = y \wedge y = z) \Rightarrow x = z) & && \text{(transitivity, E3)} \end{aligned}$$

Let τ be a signature. For every function symbol $f \in \tau$ of arity n , the following *congruence condition for functions* is valid.

$$\begin{aligned} \forall x_1, \dots, x_n, y_1, \dots, y_n \left(\bigwedge_{i=1}^n x_i = y_i \right) \\ \Rightarrow f(x_1, \dots, x_n) = f(y_1, \dots, y_n) \end{aligned} \quad \text{(E4)}$$

For every relation symbol $R \in \tau$ of arity n the following *congruence condition for relations* is valid.

$$\begin{aligned} \forall x_1, \dots, x_n, y_1, \dots, y_n \left(\bigwedge_{i=1}^n x_i = y_i \right) \\ \Rightarrow (R(x_1, \dots, x_n) \Leftrightarrow R(y_1, \dots, y_n)) \end{aligned} \quad \text{(E5)}$$

LEMMA 5.1.1. *Let $\phi(\bar{y})$ be a τ -formula and let x be a variable. Then the following sentence is valid.*

$$\forall \bar{y}. \phi \Rightarrow \forall \bar{y}, x. \phi \quad \text{(Q1)}$$

EXAMPLE 24. For a unary relation symbol R , the following $\{R\}$ -sentences

$$\begin{aligned}\forall y. R(y) &\Rightarrow \forall y, x. R(y) \\ \forall y. R(y) &\Rightarrow \forall x, y. R(y)\end{aligned}$$

are *both* covered by (Q1): the first statement can be obtained by instantiating ϕ with $R(y)$, the second can be obtained by instantiating ϕ with $\forall y. R(y)$ (note that in this case, the tuple \bar{y} in (Q1) has length 0). \triangle

Note that $\forall x. \neg\phi$ is by definition the same as $\neg\exists x. \neg\neg\phi$, which is equivalent to $\neg\exists x. \phi$. This implies the following lemma.

LEMMA 5.1.2. *Let $\phi(\bar{y}, x)$ be a τ -formula. Then the following sentence is valid.*

$$\forall \bar{y} (\forall x. \neg\phi \Rightarrow \neg\exists x. \phi) \quad (\text{Q2})$$

DEFINITION 5.1.3 (Substitution in terms). Let s and t be τ -terms and x a variable. We write $s[x \mapsto t]$ for the τ -term obtained from s by replacing all occurrences of x by the term t .

Substitution in formulas is similar, but there is an important complication.

DEFINITION 5.1.4 (Substitution in formulas). If ϕ is a τ -formula, t is a τ -term, and x is a free variable in ϕ , then we write $\phi[x \mapsto t]$ for the τ -formula obtained from ϕ by replacing all the free occurrences of x by t ; however, if one of the variables of t would be quantified by a quantifier of ϕ , then we first rename the variable of ϕ that is quantified so that this does not happen.

EXAMPLE 25. Let ϕ be the formula $\forall y. P(x, y)$ and let t be the term $f(y)$. If we simply replace x by $f(y)$ we would obtain $\forall y. P(f(y), y)$. But this is not what we want; we have to first rename y to z , and then obtain $\forall z. P(f(y), z)$. Note that this distinction has an effect on the free variables of the resulting formula. \triangle

LEMMA 5.1.5. *Let $\phi(\bar{y}, x)$ be a τ -formula and let $t(\bar{y})$ be a τ -term. Then the following sentence is valid.*

$$\forall \bar{y} (\phi[x \mapsto t] \Rightarrow \exists x. \phi) \quad (\text{Q3})$$

PROOF. Let $\bar{y} = (y_1, \dots, y_n)$. Suppose that there is a τ -structure \underline{A} and $\bar{a} \in A^n$ such that $\underline{A} \models \phi[x \mapsto t](\bar{a})$. Then $t^{\underline{A}}(\bar{a})$ provides a witness for x that shows that $\underline{A} \models \exists x. \phi(\bar{a})$. \square

LEMMA 5.1.6. *Let $\phi(\bar{y})$ and $\psi(\bar{y})$ be τ -formulas and suppose that the variable x is not free in ϕ . Then the following sentences are valid.*

$$\forall \bar{y} (\forall x (\phi \Rightarrow \psi) \Rightarrow (\phi \Rightarrow \forall x. \psi)) \quad (\text{Q4})$$

$$\forall \bar{y} ((\phi \Rightarrow \forall x. \psi) \Rightarrow \forall x (\phi \Rightarrow \psi)) \quad (\text{Q5})$$

PROOF. Let \underline{A} be a τ -structure, let $\bar{y} = (y_1, \dots, y_n)$, and let $\bar{a} \in A^n$. Then

$$\begin{aligned}\underline{A} &\models \forall x (\phi \Rightarrow \psi)(\bar{a}) \\ &\text{if and only if } \underline{A} \models (\phi \Rightarrow \psi)(\bar{a}, b) \text{ for every } b \in A \\ &\text{if and only if } \underline{A} \models \phi(\bar{a}) \Rightarrow \psi(\bar{a}, b) \text{ for every } b \in A \\ &\text{if and only if } \underline{A} \models (\phi \Rightarrow \forall x \psi)(\bar{a}).\end{aligned}$$

This shows both statements. \square

LEMMA 5.1.7. *Let ϕ be a propositional tautology with propositional variables X_1, \dots, X_m and let ψ_1, \dots, ψ_m be first-order τ -formulas with free variables x_1, \dots, x_n . Then the sentence*

$$\forall x_1, \dots, x_n. \phi(\psi_1, \dots, \psi_m) \quad (\text{TAUT})$$

is valid where $\phi(\psi_1, \dots, \psi_m)$ is the τ -formula obtained from ϕ by replacing X_i by ψ_i , for all $i \in \{1, \dots, m\}$.

PROOF. Clear. □

Our logical axioms will be (E1)-(E5), (Q1)-(Q5), and (TAUT).

5.2. Formal Proofs

Our notion of a formal proof has only one type of deduction step: modus ponens. The following is clear.

LEMMA 5.2.1 (Modus Ponens (MP)). *Let $\phi(\bar{x}), \psi(\bar{x})$ be τ -formulas and T a τ -theory. If $T \models \forall \bar{x}. \phi$ and $T \models \forall \bar{x} (\phi \Rightarrow \psi)$, then $T \models \forall \bar{x}. \psi$.*

It will be convenient to define the concept of a formal proof with respect to some fixed first-order τ -theory T ; the idea is that the sentences from T can be used like additional axioms in the proof. If $T = \emptyset$ we simply drop the reference to T in the notation. Note that in the formulas $\phi(\bar{x}), \psi(\bar{x})$ in Modus Ponens and all logical axioms the variable vector \bar{x} might also have length zero!

DEFINITION 5.2.2. Let ϕ be a τ -sentence and T a τ -theory. A *formal proof of ϕ in T* is a sequence of τ -sentences (ϕ_1, \dots, ϕ_n) with $\phi_n = \phi$ such that for every $i \in \{1, \dots, n\}$

- $\phi_i \in T$, or
- ϕ_i is a logical axiom, or
- ϕ_i can be deduced from some ϕ_j and ϕ_k with $j, k < i$ by MP, i.e.,

ϕ_j is of the form $\forall \bar{x}. \psi_j$.

ϕ_k is of the form $\forall \bar{x} (\psi_j \Rightarrow \psi_i)$,

ϕ_i is of the form $\forall \bar{x}. \psi_i$.

If there exists a formal proof of ϕ in T , then we write $T \vdash_\tau \phi$, and otherwise we write $T \not\vdash_\tau \phi$. If T is empty, we simply omit T in this notation, and write $\vdash_\tau \phi$.

REMARK 5.2.3. Note that a priori, the provability of a τ -sentence from a τ -theory T could depend on the signature allowed in the proofs, because if we add symbols to τ , there are more proofs; this is why we write $T \vdash_\tau \phi$ to indicate that only symbols from τ are allowed in the proof. In contrast, it is easy to see that the models relation \models does not depend on the signature (note that here we use the assumption that structures have non-empty domains). It will therefore be a consequence of the completeness theorem (Theorem 5.4.1) that choosing a larger signature does not increase the set of τ -formulas that can be proved from T . This is why we later simply write $T \vdash \phi$ without reference to the signature.

LEMMA 5.2.4 (Soundness). *If $T \vdash_\tau \phi$ then $T \models \phi$.*

PROOF. Clearly, if $\phi \in T$, then $T \models \phi$. If ϕ is a logical axiom, then $T \models \phi$ (Section 5.1). The statement now follows from Lemma 5.2.1 by induction over the length of the proof. □

REMARK 5.2.5. It is straightforward to write a computer program that checks for a given string whether it is a formal proof. If we require that the logical axioms obtained from a propositional tautology ϕ , (TAUT), are presented together with a proof of ϕ (e.g., a resolution refutation of a transformation of $\neg\phi$ into CNF), then we can find a program whose running time is polynomial in the length of the string.

REMARK 5.2.6. Our notion of formal proof is an example of a family of deduction systems commonly referred to as a *Hilbert-style deductive system*. Such systems might vary in the precise choice of the deduction rules and the axioms; generally speaking, fewer axioms require more deduction rules, and fewer deduction rules require more axioms.

There are also different deductive systems, typically with the goal to allow for shorter proofs or to make the respective proofs easier to read for humans, like for example Gentzen's *sequent calculus*.

REMARK 5.2.7. There is also a (sound and complete) proof system for propositional logic which exclusively uses modus ponens as a proof step, and only requires six axiom schemes, called *Frege's propositional calculus*. So we might alternatively replace (TAUT) in our notion of formal proof by the Frege axioms.

Examples of formal proofs can be found in the proofs of the following lemmata.

LEMMA 5.2.8. *Let $\phi(\bar{y}, x)$ be a τ -formula and let $t(\bar{y})$ be a τ -term. Then*

$$\vdash_{\tau} \forall \bar{y} (\forall x. \phi \Rightarrow \phi[x \mapsto t]).$$

PROOF. The following is a formal proof of $\forall \bar{y} (\forall x. \phi \Rightarrow \phi[x \mapsto t])$.

- 1 $\forall \bar{y} (\neg\phi[x \mapsto t] \Rightarrow \exists x. \neg\phi)$ (Q3)
- 2 $\forall \bar{y} ((\neg\phi[x \mapsto t] \Rightarrow \exists x. \neg\phi) \Rightarrow (\forall x. \phi \Rightarrow \phi[x \mapsto t]))$ (TAUT)
- 3 $\forall \bar{y} (\forall x. \phi \Rightarrow \phi[x \mapsto t])$ (MP(1,2)) \square

LEMMA 5.2.9 (Variable renaming lemma). *Let $\phi(\bar{y}, x)$ be a τ -formula and suppose that z is a variable that does not occur in ϕ . Then*

$$\vdash_{\tau} \forall \bar{y} (\forall z. \phi[x \mapsto z] \Rightarrow \forall x. \phi).$$

PROOF. Let $\psi := \phi[x \mapsto z]$. Note that $\psi[z \mapsto x] = \phi$. We then have the following formal proof.

- 1 $\forall \bar{y} (\forall x (\forall z. \psi \Rightarrow \psi[z \mapsto x]))$ (Lemma 5.2.8)
- 2 $\forall \bar{y} (\forall x (\forall z. \phi[x \mapsto z] \Rightarrow \phi) \Rightarrow (\forall z. \phi[x \mapsto z] \Rightarrow \forall x. \phi))$ (Q4)
- 3 $\forall \bar{y} (\forall z. \phi[x \mapsto z] \Rightarrow \forall x. \phi)$ (MP(1,2)) \square

REMARK 5.2.10. Some authors who use Hilbert-style proof systems need another deduction step, *generalisation*, and work with formulas instead of sentences, which we avoid here.

Exercises.

- (61) Show that if $T \vdash_{\tau} \forall \bar{x}. \phi$ and $T \vdash_{\tau} \forall \bar{x}. \psi$ then $T \vdash_{\tau} \forall \bar{x}. (\phi \wedge \psi)$.
- (62) Prove that if $T \vdash_{\tau} \forall \bar{y}. (\phi \Rightarrow \psi)$ and x is not free in ψ , then

$$T \vdash_{\tau} \forall \bar{y}. (\exists x. \phi \Rightarrow \psi) \quad (\exists\text{-Einführung}).$$
- (63) Prove that if $T \vdash_{\tau} \forall x. \phi$ and x is not free in ϕ , then $T \vdash_{\tau} \phi$ (\forall -Elimination).
- (64) Prove that if $T \vdash_{\tau} \forall x. (\phi \Rightarrow \psi)$ and x is not free in ϕ , then $T \vdash_{\tau} \phi \Rightarrow \forall x. \psi$ (\forall -Einführung).
- (65) Prove that if $T \vdash_{\tau} \forall \bar{y}. (\phi_1 \Rightarrow \phi_2)$ and $T \vdash_{\tau} \forall \bar{y}. (\phi_2 \Rightarrow \phi_3)$ then $T \vdash_{\tau} \forall \bar{y}. (\phi_1 \Rightarrow \phi_3)$.

5.3. Consistency

A τ -theory T is called *inconsistent* if there is a τ -sentence ϕ such that $T \vdash_{\tau} \phi$ and $T \vdash_{\tau} \neg\phi$, and it is called *consistent* otherwise. Clearly, every satisfiable theory is consistent.

LEMMA 5.3.1. *A τ -theory T is inconsistent if and only if $T \vdash_{\tau} \phi$ for any τ -sentence ϕ .*

PROOF. Let ϕ be a τ -sentence and suppose that ψ is a τ -sentence such that $T \vdash_{\tau} \psi$ and $T \vdash_{\tau} \neg\psi$. Note that $(X \Rightarrow (\neg X \Rightarrow Y))$ is a tautology, so use (TAUT) and two times (MP) to obtain $T \vdash_{\tau} \phi$. The converse implication is immediate. \square

LEMMA 5.3.2. *If $T \vdash_{\tau} \phi$ then there exists a finite subset T_0 of T such that $T_0 \vdash_{\tau} \phi$.*

PROOF. Formal proofs are finite. \square

COROLLARY 5.3.3. *Let T be a τ -theory such that all finite subsets of T are consistent. Then T is consistent as well.*

LEMMA 5.3.4 (Deduction Lemma). *Let χ and ϕ be τ -sentences and T a τ -theory. Then*

$$T \cup \{\chi\} \vdash_{\tau} \phi \text{ if and only if } T \vdash_{\tau} (\chi \Rightarrow \phi).$$

PROOF. Clearly, if $T \vdash_{\tau} (\chi \Rightarrow \phi)$ then $T \cup \{\chi\} \vdash_{\tau} \phi$ by (MP). Conversely, let (ϕ_1, \dots, ϕ_n) be a formal proof of ϕ in $T \cup \{\chi\}$. We prove by induction on $i \in \{1, \dots, n\}$ that $T \vdash_{\tau} (\chi \Rightarrow \phi_i)$.

- If $\phi_i = \chi$ this follows from (TAUT).
- If ϕ_i is from T or a logical axiom, then the statement follows from the fact that $(\phi_i \Rightarrow (\chi \Rightarrow \phi_i))$ is a tautology and (MP).
- Otherwise, ϕ_i is deduced by (MP) applied to ϕ_j and ϕ_k for $j, k < i$. Suppose that ϕ_j is of the form $\forall \bar{y}. \psi_j$, and ϕ_i is of the form $\forall \bar{y}. \psi_i$, and that ϕ_k equals $\forall \bar{y}. (\psi_j \Rightarrow \psi_i)$. By the inductive assumption we have that $T \vdash_{\tau} (\chi \Rightarrow \phi_j)$ and $T \vdash_{\tau} (\chi \Rightarrow \forall \bar{y}. (\psi_j \Rightarrow \psi_i))$. By (Q5) and (MP) we obtain that $T \vdash_{\tau} \forall \bar{y}. (\chi \Rightarrow (\psi_j \Rightarrow \psi_i))$. Then we use the fact that

$$(\chi \Rightarrow \psi_j) \Rightarrow ((\chi \Rightarrow (\psi_j \Rightarrow \psi_i)) \Rightarrow (\chi \Rightarrow \psi_i))$$

is a tautology and two times (MP) to derive that $T \vdash_{\tau} \forall \bar{y}. (\chi \Rightarrow \psi_i)$. Finally, by (Q4) and (MP) we obtain $T \vdash_{\tau} (\chi \Rightarrow \forall \bar{y}. \psi_i)$, i.e., $T \vdash_{\tau} (\chi \Rightarrow \phi_i)$. \square

COROLLARY 5.3.5. *Let T be a τ -theory and ϕ a τ -sentence. Then $T \vdash_{\tau} \phi$ if and only if $T \cup \{\neg\phi\}$ is inconsistent.*

PROOF. If $T \vdash_{\tau} \phi$, then in particular $T \cup \{\neg\phi\} \vdash_{\tau} \phi$, but clearly $T \cup \{\neg\phi\} \vdash_{\tau} \neg\phi$, so $T \cup \{\neg\phi\}$ is inconsistent.

Conversely, if $T \cup \{\neg\phi\}$ is inconsistent, then $T \cup \{\neg\phi\} \vdash_{\tau} \phi$ by Lemma 5.3.1. It follows from the deduction lemma (Lemma 5.3.4) that $T \vdash_{\tau} (\neg\phi \Rightarrow \phi)$. Since $(\neg\phi \Rightarrow \phi) \Rightarrow \phi$ is a tautology, we may use (TAUT) and (MP) to deduce that $T \vdash_{\tau} \phi$. \square

Exercises.

- (66) Let $\tau := \{R, S\}$ where R and S are unary relation symbols. Show the following (without using the completeness theorem).

$$\{\exists x.R(x), \forall y(R(y) \Rightarrow S(y))\} \vdash_{\tau} \exists x.S(x)$$

- (67) Let $\tau := \{R\}$ where R is a binary relation symbol. Find a formal proof for the following sentence.

$$\forall x, y. R(x, y) \Rightarrow \forall y, x. R(x, y).$$

- (68) Find a formal proof for the following sentence.

$$\exists x \forall y. R(x, y) \Rightarrow \forall y \exists x. R(x, y)$$

Solution proposal.

- (a) $T \vdash \forall x, y (R(x, y) \Rightarrow \exists x.R(x, y))$ (Q3)
 (b) $T \vdash \forall x, y (\forall y.R(x, y) \Rightarrow R(x, y))$ (Lemma 5.2.8)
 (c) $T \vdash \forall x (\forall y.R(x, y) \Rightarrow \exists x.R(x, y))$ (from previous two lines by Exercise 65)
 (d) $T \vdash \forall x (\forall y.R(x, y) \Rightarrow \forall y \exists x.R(x, y))$ (from previous line by Exercise 64 (\forall -Einführung))
 (e) $T \vdash \forall x (\exists x \forall y.R(x, y) \Rightarrow \forall y \exists x.R(x, y))$ (from previous line with Exercise (62) (\exists -Einführung))
 (f) $T \vdash \exists x \forall y.R(x, y) \Rightarrow \forall y \exists x.R(x, y)$ (from previous line with Exercise (63) (\forall -Elimination)).
 (69) (*) Find a formal proof in ZF that shows that the pairing axiom implies

$$\forall a \exists c (x \in c \Leftrightarrow x = a).$$

- (70) (*) Give a formal proof that the pairing axiom and the axiom of foundation imply $\forall a. \neg(a \in a)$.

REMARK 5.3.6. The previous two exercises illustrate the severe complications we are facing with writing formal proofs in our proof system for some simple mathematical facts. In the light of these exercises it is surprising how easy the proof of the completeness theorem in the next section is.

5.4. Henkin Theories

In this section, we prove the completeness theorem of first-order logic, first proved by Gödel [10].

THEOREM 5.4.1 (Completeness). *Let T be a τ -theory and let ϕ be a τ -sentence. Then $T \models \phi$ if and only if $T \vdash_{\tau} \phi$.*

It is easy to see that the completeness theorem is equivalent to the following theorem, which we will prove at the end of this section.

THEOREM 5.4.2. *A theory has a model if and only if it is consistent.*

PROOF OF THEOREM 5.4.1.

$$\begin{aligned} T \not\models \phi \text{ if and only if } T \cup \{\neg\phi\} \text{ has a model} & \quad (\text{by definition}) \\ \text{if and only if } T \cup \{\neg\phi\} \text{ is consistent} & \quad (\text{by Theorem 5.4.2}) \\ \text{if and only if } T \not\vdash_{\tau} \phi & \quad (\text{by Corollary 5.3.5}). \quad \square \end{aligned}$$

The method that we use to prove Theorem 5.4.2 is due to Henkin. One of the ideas of the proof is to work with a signature that has additional new constant symbols. We first state an easy lemma concerning new constant symbols.

LEMMA 5.4.3. *Let $\psi(\bar{y}, x)$ be a τ -formula, T a τ -theory, and let c be a constant symbol not contained in τ . The the following are equivalent:*

- (1) $T \vdash_{\tau} \forall \bar{y}, x. \psi$;
- (2) $T \vdash_{\tau \cup \{c\}} \forall \bar{y}, x. \psi$;
- (3) $T \vdash_{\tau \cup \{c\}} \forall \bar{y}. \psi[x \mapsto c]$.

PROOF. (1) \Rightarrow (2) is clear since every τ -proof is a $\tau \cup \{c\}$ -proof.

(2) \Rightarrow (3). A special case of Lemma 5.2.8.

(3) \Rightarrow (1). Let ϕ_1, \dots, ϕ_m be a formal proof of $\psi[x \mapsto c]$ in the signature $\tau \cup \{c\}$. Let z be a variable that does not appear in the entire proof ϕ_1, \dots, ϕ_m . Let ϕ'_i be the τ -formula obtained from ϕ_i by replacing all occurrences of c by z . If $\phi_i \in T$, then the symbol c does not appear in ϕ_i , and thus $T \vdash_{\tau} \forall \bar{y}, z. \phi'_i$ by (Q1) and (MP). If ϕ_i is an axiom, then so is $\forall z. \phi'_i$: this is straightforward to check for (E1)-(E5), (TAUT), and (Q1),(Q2),(Q4),(Q5). For (Q3), suppose that ϕ_i is of the form $\forall \bar{y}(\delta[x \mapsto t] \Rightarrow \exists x. \delta)$. Note that in this case ϕ'_i is of the form $\forall \bar{y}(\delta'[x \mapsto t'] \Rightarrow \exists x. \delta')$ where δ' and t' are obtained from δ and t by replacing all occurrences of c by z . Then $\forall z, \bar{y}(\delta'[x \mapsto t'] \Rightarrow \exists x. \delta')$ is an instance of (Q3). If ϕ_i is obtained from ϕ_j and ϕ_k by MP, then $\forall z. \phi'_i$ is obtained from $\forall z. \phi'_j$ and $\forall z. \phi'_k$ by MP. This shows that $T \vdash_{\tau} \forall z. \phi'_m$. Note that $\phi_m = \psi[x \mapsto c]$ and that $\phi'_m = \psi[x \mapsto z]$. Using Lemma 5.2.9 and (MP) we finally deduce that $\vdash_{\tau} \forall \bar{y}, x. \psi$. \square

LEMMA 5.4.4. *Let T be a consistent τ -theory, ϕ a τ -sentence, and $c \in \tau$ a constant symbol not occurring in $T \cup \{\phi\}$. Then $T' := T \cup \{\exists x. \phi \Rightarrow \phi[x \mapsto c]\}$ is a consistent τ -theory.*

PROOF. If T' is inconsistent then $T \vdash_{\tau} (\exists x. \phi \wedge \neg \phi[x \mapsto c])$ by Corollary 5.3.5. In particular, $T \vdash_{\tau} \exists x. \phi$. The implication (3) \Rightarrow (1) of Lemma 5.4.3 implies that $T \vdash_{\tau} \forall x. \neg \phi$. We have $T \vdash \forall x. \neg \phi \Rightarrow \neg \exists x. \phi$ by (Q2) and obtain $T \vdash_{\tau} \neg \exists x. \phi$ by (MP), so T is inconsistent. \square

DEFINITION 5.4.5. Let τ be a signature and let ρ be a set of *new* constant symbols (i.e., $\rho \cap \tau = \emptyset$). A $(\tau \cup \rho)$ -theory T is called a *Henkin theory* if for every $(\tau \cup \rho)$ -formula $\phi(x)$ there is a constant $c \in \rho$ such that

$$(\exists x. \phi(x) \Rightarrow \phi(c)) \in T.$$

The elements of ρ are called *Henkin constants* of T .

If ρ is a set of constant symbols and \underline{A} is a $(\tau \cup \rho)$ -structure such that $A = \{c^{\underline{A}} \mid c \in \rho\}$ then $\text{Th}(\underline{A})$ is a (complete) Henkin theory. To formulate a converse of this observation, we introduce the concept of *finite completeness*.

DEFINITION 5.4.6. A τ -theory T is called *finitely complete* if it is consistent and for every τ -sentence ϕ

$$\text{either } T \vdash_{\tau} \phi \text{ or } T \vdash_{\tau} \neg \phi.$$

Hils and Loeser [13] simply write *complete* instead of *finitely complete*. However, we have already defined completeness. The completeness theorem implies that finite completeness is equivalent to completeness, and once we have completely¹ proved the completeness theorem we will simply write *complete* instead of *finitely complete*.

LEMMA 5.4.7. *Every finitely complete Henkin $(\tau \cup \rho)$ -theory T with Henkin constants ρ has a model \underline{A} such that*

$$A = \{c^{\underline{A}} \mid c \in \rho\}.$$

¹I could not resist to use this word here; here, ‘completely’ is part of the meta language.

PROOF. Replacing T by the set of $(\tau \cup \rho)$ -sentences ϕ such that $T \vdash_{\tau \cup \rho} \phi$ does not change the assumptions. We may thus assume that T is *deductively closed*, that is, $T \vdash_{\tau \cup \rho} \phi$ if and only if $\phi \in T$.

We define for $c, d \in \rho$ the relation $c \sim d$ iff $(c = d) \in T$. It follows from (E1)-(E3) that \sim is an equivalence relation. We define a $(\tau \cup \rho)$ -structure \underline{A} on $A := \rho / \sim$ by setting

$$([c_1]_{\sim}, \dots, [c_k]_{\sim}) \in R^{\underline{A}} \text{ iff } R(c_1, \dots, c_k) \in T \text{ for } R \in \tau$$

$$f^{\underline{A}}([c_1]_{\sim}, \dots, [c_k]_{\sim}) = [c_0]_{\sim} \text{ iff } f(c_1, \dots, c_k) = c_0 \in T \text{ for } f \in (\tau \cup \rho).$$

It follows from the equality axioms (E4) and (E5) that this is well-defined. Also note that f is defined on all of A^k . Indeed, $(\forall \bar{y}, x. x = x) \in T$ by (E1) and (Q1), and therefore $(\forall \bar{y}. f(\bar{y}) = f(\bar{y})) \in T$ by Lemma 5.2.8. Thus, $(\forall \bar{y} \exists x. x = f(\bar{y})) \in T$ by (Q3) and $(\exists x. x = f(c_1, \dots, c_k)) \in T$ again by Lemma 5.2.8. Since T admits Henkin witnesses in ρ , there exists $c_0 \in \rho$ such that $c_0 = f(c_1, \dots, c_k) \in T$.

We claim that $\text{Th}(\underline{A}) = T$, and show by induction on the number of symbols in a first-order $(\tau \cup \rho)$ -sentences ϕ that

$$\underline{A} \models \phi \text{ if and only if } \phi \in T.$$

- ϕ is atomic. If ϕ has the form $c = d$ or $R(c_1, \dots, c_n)$, for $c, d, c_1, \dots, c_n \in \rho$, then the statement follows from the construction of \underline{A} . Otherwise, ϕ contains a function symbol $f \in \tau$, so ϕ can be written as $\psi(f(c_1, \dots, c_k))$ for some $(\tau \cup \rho)$ -formula $\psi(x)$ and $c_1, \dots, c_k \in \rho$ (note that k might be 0, in which case f is a constant symbol from τ). By construction, there exists $c \in \rho$ such that T contains $f(c_1, \dots, c_k) = c$, so $\underline{A} \models f(c_1, \dots, c_k) = c$. Thus,

$$\begin{aligned} \underline{A} \models \phi &\text{ iff } \underline{A} \models \psi(c) \\ &\text{ iff } \psi(c) \in T \\ &\text{ iff } \phi \in T. \end{aligned}$$

Here, the second equivalence is by inductive assumption since $\psi(c)$ has less symbols than ϕ .

- ϕ is of the form $\neg\psi$. Then

$$\begin{aligned} \underline{A} \models \neg\psi &\text{ iff } \underline{A} \not\models \psi \\ &\text{ iff } \psi \notin T && \text{(by inductive assumption)} \\ &\text{ iff } \neg\psi \in T && \text{(by finite completeness).} \end{aligned}$$

- ϕ is of the form $\psi_1 \wedge \psi_2$. Then

$$\begin{aligned} \underline{A} \models \psi_1 \wedge \psi_2 & \\ \text{iff } \underline{A} \models \psi_1 \text{ and } \underline{A} \models \psi_2 & \\ \text{iff } \psi_1 \in T \text{ and } \psi_2 \in T & \quad \text{(by inductive assumption)} \\ \text{iff } \psi_1 \wedge \psi_2 \in T & \quad \text{(} T \text{ is deductively closed).} \end{aligned}$$

- ϕ is of the form $\exists x. \psi(x)$. Then

$$\begin{aligned} \underline{A} \models \exists x. \psi(x) & \\ \text{iff } \underline{A} \models \psi(c) \text{ for some } c \in \rho & \quad \text{(by construction)} \\ \text{iff } \psi(c) \in T \text{ for some } c \in \rho & \quad \text{(by inductive assumption)} \\ \text{iff } \exists x. \psi(x) \in T. & \end{aligned}$$

To see the final equivalence, note that $\psi(c) \in T$ implies that $\exists x.\psi(x) \in T$ by (Q3) and (MP). For the converse, we use the assumption that T admits Henkin witnesses in ρ .

This concludes the proof that $\underline{A} \models T$; by definition, $A = \rho/\sim = \{c^A \mid c \in \rho\}$. \square

PROPOSITION 5.4.8. *Let T be a consistent τ -theory. Then T can be extended to a finitely complete Henkin theory T^* .*

PROOF. We define an increasing sequence $\emptyset = \rho_0 \subseteq \rho_1 \subseteq \dots$ of sets of constant symbols by introducing for every $(\tau \cup \rho_i)$ -formula $\phi(x)$ a new constant symbol c_ϕ and setting

$$\rho_{i+1} := \rho_i \cup \{c_\phi \mid \phi(x) \text{ a } (\tau \cup \rho_i)\text{-formula}\}.$$

Let $\rho := \bigcup_{i \in \mathbb{N}} \rho_i$. Define $T_0 := T$, and for $i \in \mathbb{N}$ define the $(\tau \cup \rho_i)$ -theory

$$T_{i+1} := T_i \cup \{\exists x.\phi(x) \Rightarrow \phi(c_\phi) \mid \phi(x) \text{ is a } (\tau \cup \rho_i)\text{-formula}\}.$$

We prove by induction over i that T_i is consistent. For $i = 0$ this holds by assumption. For $i > 0$, it suffices by Corollary 5.3.3 to show that all finite subsets S of T_i are consistent. Also note that T_{i-1} is also consistent as a $(\tau \cup \rho_i)$ -theory, as a consequence of Lemma 5.4.3. Then Lemma 5.4.4 and an induction over the size of S show that $T_{i-1} \cup S$ is a consistent $(\tau \cup \rho_i)$ -theory.

Using the fact that the union of a chain of consistent theories is consistent, we can apply Zorn's lemma (Theorem 4.1.5) to the set of consistent $(\tau \cup \rho)$ -theories that contain $\bigcup_{i \in \mathbb{N}} T_i$, partially ordered by inclusion, and obtain a maximal consistent $(\tau \cup \rho)$ -theory T^* which contains $\bigcup_{i \in \mathbb{N}} T_i$. We show that T^* is finitely complete. Let ϕ be some $(\tau \cup \rho)$ -sentence. If $T^* \not\models_{\tau \cup \rho} \phi$, then $T^* \cup \{\neg\phi\}$ is consistent by Corollary 5.3.5. Maximality then implies that $\neg\phi \in T^*$, which finishes the proof. \square

PROOF OF THEOREM 5.4.2. If T has a model, then it is consistent. Conversely, suppose that T is consistent. By Proposition 5.4.8, T is contained in a finitely complete Henkin theory T^* . By Lemma 5.4.7, T^* has a model. \square

Exercises.

- (71) Prove Theorem 5.4.2 using the completeness theorem (Theorem 5.4.1).
- (72) Show that if τ is countable, then there is a proof of Proposition 5.4.8 and hence a proof of Theorem 5.4.1 which does not require the full power of Zorn's lemma, but can be carried out in ZF using the special case of the axiom of choice for countable sets of finite sets.

5.5. Compactness

Gödel's completeness theorem implies the compactness theorem of first-order logic, which has found many applications in mathematics, e.g. in topology, set theory, and algebra. It is one of the most often used theorems in model theory.

THEOREM 5.5.1. *A theory T is satisfiable if and only if T' is satisfiable for all finite $T' \subseteq T$.*

PROOF. The completeness theorem (Theorem 5.4.1) shows that T is satisfiable if and only if T is not inconsistent. Since proofs are finite, this is the case if and only if all finite subsets of T' of T are not inconsistent, i.e., satisfiable. \square

REMARK 5.5.2. The name *compactness theorem* comes from the fact that the compactness theorem is equivalent to the statement that the following natural topological space is compact: the space is the set $\mathcal{T}(\tau)$ of all **deductively closed** complete τ -theories, and the basic open sets are the sets T_ϕ of the form $\{T \in \mathcal{T}(\tau) \mid \phi \in T\}$.

To see the equivalence, let \mathcal{C} be a covering of $\mathcal{T}(\tau)$ by open subsets of $\mathcal{T}(\tau)$. We may assume that \mathcal{C} is of the form $\{T_\phi \mid \phi \in S\}$ for some set S of τ -sentences. Note that $S' := \{\neg\phi \mid \phi \in S\}$ is unsatisfiable. The compactness theorem of first-order logic implies that there is a finite subset F of S' which is unsatisfiable. But then $\{T_{\neg\phi} \mid \phi \in F\}$ is a finite subset of \mathcal{C} covering $\mathcal{T}(\tau)$, showing that $\mathcal{T}(\tau)$ is compact.

Conversely, suppose that $\mathcal{T}(\tau)$ is compact, and that the τ -theory S is inconsistent. Then $\{T_{\neg\phi} \mid \phi\}$ is an open covering of $\mathcal{T}(\tau)$. So by compactness it has a finite subcovering, i.e., there is a finite subset F of S such that $\bigcup\{T_{\neg\phi} \mid \phi \in F\} = \mathcal{T}(\tau)$. Hence, F is inconsistent, which is the statement of the compactness theorem. \square

REMARK 5.5.3. There are other proofs of the compactness theorem which do not rely on the completeness theorem. For example, there are proofs based on *ultraproducts*, which can be found in many model theory text books (or see [2]).

The following corollaries present well-known consequences of the compactness theorem.

COROLLARY 5.5.4. *Let T be a first-order theory with arbitrarily large finite models. Then T has an infinite model.*

PROOF. By assumption, every finite subset of

$$T' := T \cup \left\{ \exists x_1, \dots, x_k \bigwedge_{1 \leq i < j \leq k} x_i \neq x_j \mid k \in \mathbb{N} \right\}$$

has a model. By the compactness theorem, T' has a model, and every model of T' must be infinite. \square

Recall that a graph G is k -colourable if and only if it has a homomorphism to K_k , the clique with k vertices (Examples 9 and 10).

COROLLARY 5.5.5. *A graph $\underline{G} = (V; E)$ is k -colourable if and only if all its finite subgraphs are k -colourable.*

PROOF. Let τ be the signature $\{f, c_1, \dots, c_k\} \cup \{c_v \mid v \in V\}$ where f is a unary function symbol and all other symbols are distinct constant symbols. Consider the following τ -theory T .

$$T := \left\{ f(c_u) \neq f(c_v) \mid (u, v) \in E^{\underline{G}} \right\} \\ \cup \left\{ \bigwedge_{i \neq j} c_i \neq c_j, \forall x \bigvee_{i \in \{1, \dots, k\}} f(x) = c_i \right\}$$

By assumption, every finite subset of T is satisfiable. By the compactness theorem, T has a model \underline{M} , and $f^{\underline{M}}$ gives the desired k -colouring of \underline{G} . \square

An *unfriendly partition* of a graph (V, E) is a partition of the vertices $V = V_1 \uplus V_2$ such that for every $i \in \{1, 2\}$, every $x \in V_i$ has at least as many neighbours in V_{3-i} as in V_i . Every finite graph has an unfriendly partition since every partition $V = V_1 \uplus V_2$ with maximally many edges between V_1 and V_2 is unfriendly.

CONJECTURE 5.1 (Cowan and Emerson, unpublished). *Every countable graph has an unfriendly partition.*

The conjecture is false for uncountable graphs (Shelah and Milner [27]). But the conjecture is true for *locally finite* graphs: a graph $(V; E)$ is *locally finite* if for every vertex $x \in V$ the set of neighbours $N(x) := \{y \in V \mid (x, y) \in E\}$ is finite.

COROLLARY 5.5.6. *Every locally finite graph \underline{G} has an unfriendly partition.*

PROOF. Let $\{c_v \mid v \in G\}$ be a set of pairwise distinct constant symbols and P a unary relation symbol. Let $\phi_{v,n}$ be the sentence

$$\left(\bigwedge_{\text{distinct } u_1, \dots, u_n \in N(u)} \bigwedge_{i \leq n} P(c_{u_i}) \right) \Rightarrow \bigvee_{\text{distinct } v_1, \dots, v_n \in N(u)} \bigwedge_{i \leq n} \neg P(c_{v_i}).$$

Analogously, there is a first-order sentence $\psi_{v,n}$ expressing that if c_v has n neighbours in \underline{G} in the complement of P , then it has in \underline{G} also n neighbours in P . Consider the theory

$$T := \{\phi_{v,n} \mid v \in G, n \in \mathbb{N}\} \cup \{\psi_{v,n} \mid v \in G, n \in \mathbb{N}\}.$$

Observe that every finite subset of T has a model, since every finite graph has an unfriendly partition (Why?!). The compactness theorem therefore implies that T has a model \underline{M} . Then $G_1 := P^{\underline{M}} \cap G$ and $G_2 := G \setminus G_1$ is an unfriendly partition of \underline{G} . \square

EXAMPLE 26. Note that every linearly ordered group \underline{G} (see Example 7) is *torsion-free*, i.e., if $a \in G$ is such that $a^m = e$ for some $m \in \mathbb{N}$, then $a = e$. Conversely, a torsion-free abelian group \underline{G} can be expanded to a linearly ordered group (Levi). To see this, let T be the theory of torsion-free ordered abelian groups; note that the definition of torsion-freeness above, and the definition of ordered abelian groups can be phrased using *universal* first-order sentences (i.e., a sentence that starts with universal quantifiers in front of a quantifier-free part), so we assume that T is universal. Let S be the set of all atomic first-order sentences that hold in \underline{G}_G (see Section 3.1.4). Note that the $\{+, -, 0, <\}$ -reduct of a model of $S \cup T$ is an ordered abelian group \underline{H} , and that G is the domain of a substructure \underline{G}' of \underline{H} . Since T is universal, \underline{G}' satisfies all sentences of T , too, and hence \underline{G}' is an expansion of \underline{G} which is an ordered abelian group, as desired. So all we have to prove is that $T \cup S$ is satisfiable. By the compactness theorem, it suffices to show that $T \cup F$ is satisfiable for every finite subset F of S . Let F be a finite subset of S . Only finitely many constant symbols can be mentioned in F ; consider the smallest subgroup \underline{S} of \underline{G} that contains the elements denoted by these constants. Then \underline{S} is a finitely generated abelian group; those groups are classified, by the *fundamental theorem for finitely generated abelian groups*. Since \underline{S} is torsion-free, they must be of the form $(\mathbb{Z}^n; +, -, 0)$. Hence, \underline{S} can be linearly ordered, for example by the *lexicographic ordering*, defined by $(x_1, \dots, x_n) < (y_1, \dots, y_n)$ if $x_1 = y_1, \dots, x_{i-1} = y_{i-1}$ and $x_i < y_i$, for some $i \in \{1, \dots, n\}$. This shows that $T \cup F$ is satisfiable. \triangle

Exercises.

- (73) Let c_1, c_2, \dots be constant symbols, and let \underline{B} be the $\{<, c_1, c_2, \dots\}$ -expansion of $(\mathbb{Q}; <)$ where $c_i^{\underline{B}} = 1/i$. Show that there is a model \underline{A} of $\text{Th}(\underline{B})$ with an element ϵ such that $\underline{A} \models \epsilon > 0$ and $\underline{A} \models \epsilon < c_i$ for every $i \in \mathbb{N}$.
- (74) Let T be a first-order theory and $\phi(x)$ a formula. Show that if T has for every $n \in \mathbb{N}$ a model \underline{A} with $|\phi^{\underline{A}}| \geq n$, then T has a model \underline{A} such that $\phi^{\underline{A}}$ is infinite.
- (75) Show that the compactness theorem does not hold if we allow infinite disjunctions as sentences. **How about infinite conjunctions?**
- (76) A (finite or infinite) set S of propositional formulas is called *satisfiable* if there is a mapping from the variables that appear in formulas from S to $\{0, 1\}$ which satisfies all formulas in S . Show that the compactness theorem for first-order logic implies the compactness theorem for propositional logic: if every finite subset of S is satisfiable, then all of S is satisfiable.

- (77) Let \underline{G} be a 2-colorable graph with color classes $A, B \subseteq G$ such that every vertex in A has only finitely many neighbours. Suppose that every finite subset A' of A has a *matching*, i.e., a subset M of the edges of \underline{G} such that any two distinct edges in M are disjoint and every $a \in A'$ appears in an edge of M . Show that then all of A has a matching.
- (78) Show that the compactness theorem is equivalent to the following statement. If a first-order theory T has the same models as a single first-order sentence ϕ , there is already a finite subset of T which has the same models as ϕ .
- (79) (Exercise 2.2.3 in [30]) A class \mathcal{C} of τ -structures is called
- *elementary* if there exists a first-order τ -theory T such that \mathcal{C} are precisely the models of T ;
 - *finitely axiomatisable* if it is the class of models of a finite theory.
- Show that \mathcal{C} is finitely axiomatisable if and only if \mathcal{C} is an elementary class and the complement of \mathcal{C} is an elementary class.
- (80) (Exercise 2.2.5 in [30]) The *characteristic* of a field is the smallest positive number n such that

$$\underbrace{1 + \cdots + 1}_{n \text{ times}} = 0,$$

if such a number exists, and 0 otherwise. Let T be a τ_{Ring} -theory containing T_{Field} (see Example 17). Show that

- if T has models of arbitrarily large characteristic, then it has a model of characteristic 0.
- The class of fields of characteristic 0 is not finitely axiomatisable.

Computability

This chapter presents the basics of the theory of computable functions (also called *recursive functions*); the results will be important for proving the incompleteness theorems in the next chapter. We follow the classical approach to first introduce the class of *primitive recursive functions*, which is already quite large, but does not contain some recursive functions that grow too fast for being primitive recursive. The Ackermann function is an example of such a function. The recursive functions extend the primitive recursive functions by so-called μ -recursion.

Recursive functions can be characterised in many different ways. A particularly useful characterisation is via an abstract machine model, *Turing machines*. Turing machines can be represented by bit strings, and one can show that there is a *universal Turing machine* that, given a string that describes a Turing machine, simulates the computation of that Turing machine for any given input. We will also present a concrete function that is not recursive, derived from the Halting problem for Turing machines. It will then be easy to also prove the first incompleteness theorem of Gödel in the next chapter.

6.1. Primitive Recursion

For $n \in \mathbb{N}$, we define $\mathcal{O}^{(n)} := \{f: \mathbb{N}^n \rightarrow \mathbb{N}\}$ and $\mathcal{O} := \bigcup_{n \in \mathbb{N}} \mathcal{O}^{(n)}$.

DEFINITION 6.1.1. A subset \mathcal{C} of \mathcal{O} is called a *clone*¹ if

- \mathcal{C} contains for every $n \in \mathbb{N}^+$ and every $i \in \{1, \dots, n\}$ the i -th projection of arity n , i.e., the operation $\pi_i^n \in \mathcal{O}^{(n)}$ defined by $(x_1, \dots, x_n) \mapsto x_i$.
- \mathcal{C} is *closed under composition*: if $f_1, \dots, f_n \in \mathcal{C} \cap \mathcal{O}^{(p)}$ and $g \in \mathcal{C} \cap \mathcal{O}^{(n)}$, then the operation $g(f_1, \dots, f_n) \in \mathcal{O}^{(p)}$ given by

$$(x_1, \dots, x_p) \mapsto g(f_1(x_1, \dots, x_p), \dots, f_n(x_1, \dots, x_p))$$

is also contained in \mathcal{C} .

We also write $\mathcal{C}^{(n)}$ for $\mathcal{C} \cap \mathcal{O}^{(n)}$.

Exercises.

- (81) Let $\mathcal{C} \subseteq \mathcal{O}$ be a clone, $f \in \mathcal{C}^{(k)}$, and $g \in \mathcal{C}^{(m)}$. Let $f * g \in \mathcal{O}^{(km)}$ be defined as follows.

$$\begin{aligned} (f * g)(x_{1,1}, \dots, x_{1,m}, x_{2,1}, \dots, x_{k,1}, \dots, x_{k,m}) \\ := f(g(x_{1,1}, \dots, x_{1,m}), \dots, g(x_{k,1}, \dots, x_{k,m})) \end{aligned}$$

Show that $f * g \in \mathcal{C}$.

¹The usual definition of clones excludes operations of arity 0; it will be convenient for us to allow them, and it does not make a big theoretical difference.

6.1.1. Primitive recursive functions. The set of *primitive recursive*² functions \mathcal{P} is the smallest subset of \mathcal{O} which satisfies the following properties:

- \mathcal{P} is a clone;
- \mathcal{P} contains the 0-ary constant operation $c_0^0 \in \mathcal{O}^{(0)}$ that is constant 0;
- \mathcal{P} contains the *successor function* $s \in \mathcal{O}^{(1)}$ defined by $x \mapsto x + 1$;
- \mathcal{P} is *closed under recursion*: if $g \in \mathcal{P}^{(n)}$ and $h \in \mathcal{P}^{(n+2)}$, then the operation $\rho(g, h) := f \in \mathcal{O}^{(n+1)}$ defined by

$$\begin{aligned} f(x_1, \dots, x_n, 0) &:= g(x_1, \dots, x_n) \\ f(x_1, \dots, x_n, y + 1) &:= h(x_1, \dots, x_n, y, f(x_1, \dots, x_n, y)) \end{aligned}$$

is also in \mathcal{P} .

Formally, \mathcal{P} is the intersection over all subsets of \mathcal{O} that have the closure properties above; the definition of \mathcal{P} makes sense because the intersection then also has the closure properties.

We use the following notation for specifying operations in \mathcal{O} : if $t(x_1, \dots, x_n)$ is a term which describes the function f of arity n that maps $(a_1, \dots, a_n) \in \mathbb{N}^n$ to $t(a_1, \dots, a_n)$, then f is denoted by $\lambda x_1, \dots, x_n. t$. For example, the i -th projection of arity n equals $\lambda x_1, \dots, x_n. x_i$, and the successor function equals $\lambda x. x + 1$.

LEMMA 6.1.2. *The operations $\lambda x, y. x + y$ and $\lambda x, y. x \cdot y$ are primitive recursive.*

PROOF. Let $h := s(\pi_3^3)$ and $g = \pi_1^1$. Then $\rho(g, h) = \lambda x, y. x + y$, because $x + 0 = x$ and $x + (y + 1) = (x + y) + 1$. The proof for $\lambda x, y. x \cdot y$ is similar. \square

DEFINITION 6.1.3. The operation $\dot{-} : \mathbb{N}^2 \rightarrow \mathbb{N}$ is defined as follows.

$$x \dot{-} y := \begin{cases} x - y & \text{if } x \geq y \\ 0 & \text{otherwise.} \end{cases}$$

LEMMA 6.1.4. *The operation $\dot{-}$ is primitive recursive.*

PROOF. We first construct $\lambda y. y \dot{-} 1$ using $0 \dot{-} 1 = 0$ and $(y + 1) \dot{-} 1 = y$ by recursion. Then $\dot{-}$ can be constructed using $x \dot{-} 0 = x$ and $x \dot{-} (y + 1) = (x \dot{-} y) \dot{-} 1$ by recursion. \square

LEMMA 6.1.5. *If $f \in \mathcal{P}^{(n+1)}$, then $\lambda x_1, \dots, x_n, y. \sum_{t=0}^y f(\bar{x}, t) \in \mathcal{P}^{(n+1)}$.*

PROOF. By recursion. \square

Exercises.

- (82) Show that for every constant $k \in \mathbb{N}$ the operation $c_k^n := \lambda x_1, \dots, x_n. k \in \mathcal{O}^{(n)}$ that always returns k is primitive recursive.
- (83) Show that the smallest clone that contains s and c_0^0 is not closed under recursion.
- (84) Show that $\lambda x. x!$ and $\lambda x, y. x^y$ are primitive recursive.

6.1.2. Primitive recursive sets. If $X \subseteq \mathbb{N}^n$ then the *characteristic function* for X , denoted by 1_X , is the function defined by

$$1_X(x_1, \dots, x_n) := \begin{cases} 1 & \text{if } (x_1, \dots, x_n) \in X; \\ 0 & \text{else.} \end{cases}$$

A subset X of \mathbb{N}^n is called *primitive recursive* if its characteristic function 1_X is primitive recursive. For every $n \in \mathbb{N}$, the set \mathbb{N}^n is primitive recursive (Exercise 82).

²Before the discovery of operations in \mathcal{O} that are computable but not primitive recursive (see Section 6.1.5) *recursive* was used instead of *primitive recursive*.

LEMMA 6.1.6. *The set $\{(x, y) \mid x < y\} \subseteq \mathbb{N}^2$ is primitive recursive.*

PROOF. We have $1_{<}(x, y) = 1 \dot{-} (1 \dot{-} (y \dot{-} x))$. \square

LEMMA 6.1.7. *The set of primitive recursive subsets of \mathbb{N}^n is closed under \cup , \cap , and complementation.*

PROOF. Let X be a primitive recursive subset of \mathbb{N}^n . Then $1_{\mathbb{N}^n \setminus X} = 1 \dot{-} 1_X$, and hence $\mathbb{N}^n \setminus X$ is primitive recursive. If X and Y are primitive recursive subsets of \mathbb{N}^n , then $1_{X \cap Y} = 1_X \cdot 1_Y$ and hence $X \cap Y$ is primitive recursive; this proves the lemma since $X \cup Y = \mathbb{N}^n \setminus ((\mathbb{N}^n \setminus X) \cap (\mathbb{N}^n \setminus Y))$. \square

LEMMA 6.1.8 (Definition by cases). *Let $\mathbb{N}^n = A_1 \uplus \cdots \uplus A_k$ be a partition of \mathbb{N}^n into primitive recursive sets, and let $f_1, \dots, f_k \in \mathcal{P}^{(n)}$. Then the function f defined by $f(\bar{x}) := f_i(\bar{x})$ if $\bar{x} \in A_i$, for all $i \in \{1, \dots, k\}$, is primitive recursive.*

PROOF. One has $f = 1_{A_1} \cdot f_1 + \cdots + 1_{A_k} \cdot f_k$. \square

We will see later (Exercise 100) that if $X \subseteq \mathbb{N}^{n+1}$ is primitive recursive, then

$$\{(x_1, \dots, x_n) \in \mathbb{N}^n \mid \text{there exists } x_{n+1} \text{ such that } (x_1, \dots, x_n, x_{n+1}) \in X\}$$

need not be primitive recursive. However, the following lemma shows that the primitive recursive sets are closed under a bounded version of quantification.

LEMMA 6.1.9. *If $X \subseteq \mathbb{N}^{n+1}$ is a primitive recursive set, so are the sets*

$$X_{\exists} := \{(x_1, \dots, x_n, z) \in \mathbb{N}^{n+1} \mid \text{there exists } t \leq z \text{ such that } (\bar{x}, t) \in X\}$$

$$\text{and } X_{\forall} := \{(x_1, \dots, x_n, z) \in \mathbb{N}^{n+1} \mid \text{for all } t \leq z \text{ we have } (\bar{x}, t) \in X\}.$$

PROOF. Since $X_{\forall} = \mathbb{N}^{n+1} \setminus (\mathbb{N}^{n+1} \setminus X)_{\exists}$ and by Lemma 6.1.7, it is enough to treat the first case. The operation $\lambda x_1, \dots, x_n, z \sum_{t=0}^z 1_X(\bar{x}, t)$ is primitive recursive by Lemma 6.1.5. Note that $1_{X_{\exists}}(\bar{x}, z) = 1$ if $\sum_{t=0}^z 1_X(\bar{x}, t) \geq 1$ and $1_{X_{\exists}}(\bar{x}, z) = 0$ otherwise. Hence, the statement follows by Lemma 6.1.8. \square

Exercises.

(85) Show that the binary operation $1_{=}$ is primitive recursive.

(86) Prove that the set of primitive recursive subsets of \mathbb{N} is not closed under infinite intersections and unions.

6.1.3. Bounded μ -recursion. Let $X \subseteq \mathbb{N}^{n+1}$. Then the function $f: \mathbb{N}^{n+1} \rightarrow \mathbb{N}$ defined by

$$f(\bar{x}, z) := \begin{cases} 0 & \text{if there is no } t \leq z \text{ with } (\bar{x}, t) \in X \\ t & \text{if } t \in \{0, \dots, z\} \text{ is smallest such that } (\bar{x}, t) \in X \end{cases}$$

is denoted by

$$(\mu t \leq z)X(\bar{x}, t).$$

LEMMA 6.1.10. *Let $X \subseteq \mathbb{N}^{n+1}$ be primitive recursive. Then $f := (\mu t \leq z)X(\bar{x}, t)$ is primitive recursive as well.*

PROOF. We have $f(\bar{x}, 0) = 0$ and

$$f(\bar{x}, z+1) = \begin{cases} f(\bar{x}, z) & \text{if } \sum_{t=0}^z 1_X(\bar{x}, t) \geq 1, \\ z+1 & \text{if } \sum_{t=0}^z 1_X(\bar{x}, t) = 0 \text{ and } (\bar{x}, z+1) \in X, \\ 0 & \text{otherwise.} \end{cases}$$

The statement then follows from Lemma 6.1.8 and recursion. \square

EXAMPLE 27. Let $q: \mathbb{N}^2 \rightarrow \mathbb{N}$ be the function which maps (x, y) to the integer part of x/y if $y \neq 0$, and to 0 otherwise. Then q is primitive recursive; indeed, $q(x, y) = (\mu t \leq x)((t+1) \cdot y > x)$. \triangle

6.1.4. Gödel numbers. To formulate more complex functions we need a coding trick, due to Gödel. We start with the easy observation that pairs of natural numbers can be coded by a single natural number, in the following sense.

LEMMA 6.1.11. *There exists a bijection $\alpha_n: \mathbb{N}^n \rightarrow \mathbb{N}$ which is primitive recursive. Moreover, there are primitive recursive functions $\beta_1^n, \dots, \beta_n^n: \mathbb{N} \rightarrow \mathbb{N}$ such that the function $x \mapsto \alpha_n(\beta_1^n(x), \dots, \beta_n^n(x))$ is the identity on \mathbb{N} .*

PROOF. We show the statement for $n = 2$; the general case can be shown by induction. The function α_2 given by $(x, y) \mapsto \frac{(x+y+1)(x+y)}{2} + x$ from Example 23 is clearly primitive recursive (using Example 27). Define

$$\beta_1^2 := \lambda z(\mu x \leq z)(\exists y \leq z)(\alpha_2(x, y) = z)$$

$$\text{and } \beta_2^2 := \lambda z(\mu y \leq z)(\exists x \leq z)(\alpha_2(x, y) = z).$$

Since $\alpha_2(x, y) \geq \min(x, y)$, for every $z \in \mathbb{N}$ we have $\alpha_2(\beta_1^2(z), \beta_2^2(z)) = z$. \square

The coding trick of Gödel is based on representing arbitrarily long finite sequences of natural numbers as natural numbers. We first need some preparatory lemmas.

LEMMA 6.1.12. *The divisibility relation*

$$D := \{(x, y) \in \mathbb{N}^2 : y|x\} = \{(x, y) \in \mathbb{N}^2 : \text{there exists } z \in \mathbb{N} \text{ such that } y \cdot z = x\}$$

is primitive recursive.

PROOF. First define the modulo function $m: \mathbb{N}^2 \rightarrow \mathbb{N}$ recursively by $m(0, y) = 0$ and

$$m(x+1, y) = \begin{cases} 0 & \text{if } m(x, y) + 1 = y \\ m(x, y) + 1 & \text{otherwise.} \end{cases}$$

It follows from Lemma 6.1.8 that $m \in \mathcal{P}$. Then $1_D(x, y) = 1 \dot{-} m(x, y)$. \square

LEMMA 6.1.13. *The set $P \subseteq \mathbb{N}$ of prime numbers is primitive recursive. The function $\pi: \mathbb{N} \rightarrow \mathbb{N}$ that maps $n \in \mathbb{N}$ to the n -th prime number is primitive recursive.*

PROOF. We have that x is a prime number if and only if it satisfies

$$x \geq 2 \wedge \forall y \leq x (y|x \Rightarrow (y = 1 \vee y = x)).$$

It follows from Lemma 6.1.6, Lemma 6.1.7, and bounded quantification (Lemma 6.1.9) that the set of all natural numbers that satisfy this formula is primitive recursive. For the second statement, note that $\pi(0) = 2$ and that

$$\pi(x+1) = (\mu z \leq \pi(x)! + 1)(z > \pi(x) \wedge P(z)). \quad \square$$

DEFINITION 6.1.14. Let (x_0, \dots, x_{n-1}) be a finite sequence of natural numbers. Then the *Gödel number* of (x_0, \dots, x_{n-1}) is defined to be

$$\langle x_0, \dots, x_{n-1} \rangle := \pi(0)^{x_0} \cdot \dots \cdot \pi(n-2)^{x_{n-2}} \cdot \pi(n-1)^{x_{n-1}+1} - 1.$$

EXAMPLE 28. *If $\langle \rangle$ is the tuple of length $n = 0$, then $\langle \rangle = 1 - 1 = 0$ since the product of an empty set of numbers is 1 by definition. If $n = 1$ and $x_0 = 0$, then $\langle x_0 \rangle = \pi(0)^{x_0+1} - 1 = 2^1 - 1 = 1$. If $n = 2$, $x_0 = 0 = x_1 = 1$, then $\langle x_0, x_1 \rangle = \pi(0)^{x_0} \pi(1)^{x_1+1} - 1 = 1 \cdot 3 - 1 = 2$.* \triangle

LEMMA 6.1.15. *The map $\langle \cdot \rangle$ defines a bijection between the set of finite sequences of natural numbers and \mathbb{N} , and has the following properties:*

(1) The length function ℓ defined by

$$\ell(\langle x_0, \dots, x_{n-1} \rangle) := n$$

is primitive recursive.

(2) The binary component function $(x)_i$ defined by

$$\langle x_0, \dots, x_{n-1} \rangle_i := \begin{cases} x_i & \text{if } i < n, \\ 0 & \text{else.} \end{cases}$$

is primitive recursive.

PROOF. The set

$$X := \{(x, y) \mid \neg\pi(z) \mid (x+1) \text{ for all } z \in \{y, \dots, x\}\}$$

is primitive recursive by Lemmas 6.1.12, 6.1.13, 6.1.6, 6.1.7, and 6.1.9. **Note that $\ell(x) \leq x$.** The function ℓ can be written as

$$\ell(x) = \begin{cases} 0 & \text{if } x = 0 \\ (\mu y \leq x) X(x, y) & \text{if } x > 0 \end{cases}$$

and hence is primitive recursive by Lemma 6.1.8.

To prove (2), **first observe that if $x > 0$ then $(x)_i < x$.** Consider the primitive recursive set $Y := \{(i, x, y) \mid \neg(\pi(i)^y \mid (x+1))\}$. Then the function $(x)_i$ can be written as

$$(x)_i := \begin{cases} 0 & \text{if } i + 1 > \ell(x) \\ (\mu y \leq x) Y(i, x, y + 2) & \text{if } i + 1 = \ell(x) \\ (\mu y \leq x) Y(i, x, y + 1) & \text{if } i + 1 < \ell(x) \end{cases}$$

and hence is primitive recursive by Lemma 6.1.8. \square

6.1.5. The Ackermann function. We present a famous example of an operation which is not primitive recursive, the *Ackermann function* $\xi \in \mathcal{O}^{(2)}$. It is historically one of the first discovered function of this type. The original function had three arguments; we present a modification of a formulation of Rózsa Péter that is used by Hils and Loeser [13] and by Cori and Lascar [24]; the motivation of the modification is to simplify the proof that the Ackermann function is not primitive recursive.

DEFINITION 6.1.16. The Ackermann function is defined inductively as follows.

- $\xi(0, x) := 2^x$
- $\xi(n, 0) := 1$
- $\xi(n + 1, x + 1) := \xi(n, \xi(n + 1, x))$.

This is well-defined, since the lexicographic order of \mathbb{N}^2 is a well-ordering (in the inductive step, either the first argument decreases, or the first argument remains equal, in which case the second argument decreases). See Figure 6.1 for the first terms of this function. In the following we write $\xi_n(x)$ for $\xi(n, x)$ and first prove some simple properties of $\xi_n : \mathbb{N} \rightarrow \mathbb{N}$.

LEMMA 6.1.17. *Let $n, x \in \mathbb{N}$. Then $\xi_n(x) > x$. If $y > x$, then $\xi_n(y) > \xi_n(x)$. Moreover, $\xi_{n+1}(x) \geq \xi_n(x)$.*

PROOF. We prove the first and second statement by induction on n . For $n = 0$, we have $\xi_0(x) = \xi(0, x) = 2^x > x$, and for $y > x$ we have $\xi_0(y) = 2^y > 2^x = \xi_0(x)$. Now suppose that the statements hold for n ; then they imply

$$\xi_{n+1}(x + 1) = \xi_n(\xi_{n+1}(x)) > \xi_{n+1}(x)$$

$\xi(n, x)$	0	1	2	3	4	x
0	1	2	4	2^3	2^4	
1	1	2	4	2^{2^2}	$2^{2^{2^2}}$	
2	1	2	4	$2^{2^{2^2}}$	\dots	
n						

FIGURE 6.1. The values of the Ackerman function $\xi \in \mathcal{O}^{(2)}$ for small arguments.

which shows the second statement for $n+1$. The first statement for $n+1$ follows from the second statement: $\xi_{n+1}(x) > \xi_{n+1}(x-1) > \dots > \xi_{n+1}(0) = 1$, so $\xi_{n+1}(x) > x$.

To prove the third statement, note that $\xi_{n+1}(0) = 1 = \xi_n(0)$ and that

$$\begin{aligned} \xi_{n+1}(x+1) &= \xi_n(\xi_{n+1}(x)) && \text{(by the definition of } \xi_{n+1}\text{)} \\ &\geq \xi_n(x+1) && \text{(by the second statement)} \end{aligned}$$

where the second statement of the lemma can be applied because $\xi_{n+1}(x) > x$ by the first statement. \square

Define $\xi_n^k := \underbrace{\xi_n \circ \dots \circ \xi_n}_{k \text{ times}}$ for $k \geq 1$ and $\xi_n^0(x) := x$.

LEMMA 6.1.18. *For all $k, n, x \in \mathbb{N}$ we have $\xi_n^k(x) \leq \xi_{n+1}(x+k)$.*

PROOF. Our proof is by induction on k . The base case $k = 0$ follows from Lemma 6.1.17. Now suppose that the result holds for k . Then

$$\begin{aligned} \xi_n^{k+1}(x) &= \xi_n(\xi_n^k(x)) \leq \xi_n(\xi_{n+1}(x+k)) && \text{(by inductive assumption)} \\ &= \xi_{n+1}(x+k+1). && \square \end{aligned}$$

Note that for each fixed n , the function ξ_n is primitive recursive (Exercise 88). However, the function ξ is not. To prove this, we need a better understanding of the maximal growth of primitive recursive functions.

DEFINITION 6.1.19. A function $f \in \mathcal{O}^{(1)}$ *dominates* a function $g \in \mathcal{O}^{(m)}$ if there exists $\ell \in \mathbb{N}$ such that for all $(x_1, \dots, x_m) \in \mathbb{N}^m$

$$g(x_1, \dots, x_m) \leq f(\max(x_1, \dots, x_m, \ell)).$$

For $n \in \mathbb{N}$, we denote by \mathcal{B}_n the set of all operations in \mathcal{O} that are dominated by ξ_n^k for at least one $k \in \mathbb{N}$.

LEMMA 6.1.20. *For every $n \in \mathbb{N}$ the set \mathcal{B}_n is a clone.*

PROOF. Let $f_1, \dots, f_m \in \mathcal{B}_n^{(k)}$ and $g \in \mathcal{B}_n^{(m)}$. Then there are $\ell_0, \ell_1, \dots, \ell_m, k_0, k_1, \dots, k_m \in \mathbb{N}$ such that for all $x_1, \dots, x_n \in \mathbb{N}$ and $i \in \{1, \dots, m\}$

$$\begin{aligned} g(x_1, \dots, x_m) &\leq \xi_n^{k_0}(\max(x_1, \dots, x_m, \ell_0)) \\ \text{and } f_i(x_1, \dots, x_k) &\leq \xi_n^{k_i}(\max(x_1, \dots, x_k, \ell_i)). \end{aligned}$$

Define $\ell := \max(\ell_0, \ell_1, \dots, \ell_m)$ and $k := k_0 + \max(k_1, \dots, k_m)$. Let $\bar{x} \in \mathbb{N}^k$. Then

$$\begin{aligned} g(f_1(\bar{x}), \dots, f_m(\bar{x})) &\leq g(\xi_n^{k_1}(\max(\bar{x}, \ell_1)), \dots, \xi_n^{k_m}(\max(\bar{x}, \ell_m))) \\ &\leq \xi_n^{k_0}(\max(\xi_n^{k_1}(\max(\bar{x}, \ell_1)), \dots, \xi_n^{k_m}(\max(\bar{x}, \ell_m)), \ell_0)) \\ &\leq \xi_n^{k_0}(\xi_n^{\max(k_1, \dots, k_m)}(\max(\bar{x}, \ell_0, \ell_1, \dots, \ell_m))) \\ &\leq \xi_n^k(\max(\bar{x}, \ell)) \end{aligned}$$

which proves that \mathcal{B}_n is closed under composition. \square

LEMMA 6.1.21. Let $g \in \mathcal{B}_n^{(p)}$ and $h \in \mathcal{B}_n^{(p+2)}$. Then $\rho(g, h) \in \mathcal{B}_{n+1}^{(p)}$. In particular,

$$\mathcal{P} \subseteq \bigcup_{n \in \mathbb{N}} \mathcal{B}_n.$$

PROOF. By assumption, there are $k_1, \ell_1, k_2, \ell_2 \in \mathbb{N}$ such that for all $x_1, \dots, x_p \in \mathbb{N}$

$$\begin{aligned} g(\bar{x}) &\leq \xi_n^{k_1}(\max(x_1, \dots, x_p, \ell_1)) \\ h(\bar{x}, y, t) &\leq \xi_n^{k_2}(\max(x_1, \dots, x_p, y, t, \ell_2)). \end{aligned}$$

By induction on $y \in \mathbb{N}$ one can show that

$$\rho(g, h)(\bar{x}, y) \leq \xi_n^{k_1 + yk_2}(\max(x_1, \dots, x_p, y, \ell_1, \ell_2))$$

which is at most $\xi_{n+1}(\max(x_1, \dots, x_p, y, \ell_1, \ell_2) + k_1 + yk_2)$ by Lemma 6.1.18. The upper bound is in \mathcal{B}_{n+1} by Lemma 6.1.20, which proves the claim. \square

The following result was conjectured by Hilbert and proved by Ackermann.

THEOREM 6.1.22. The Ackermann function ξ is not primitive recursive.

PROOF. Suppose for contradiction that ξ is primitive recursive. Then $\lambda x. \xi_x(2x)$ is primitive recursive as well and by Lemma 6.1.21 there exist $n, k, \ell \in \mathbb{N}$ such that for all $x \in \mathbb{N}$ with $x \geq \ell$

$$\xi_x(2x) \leq \xi_n^k(x).$$

For $x > \max(k, n + 1, \ell)$ we thus obtain

$$\begin{aligned} \xi_{n+1}(x + k) &< \xi_x(2x) && \text{(Lemma 6.1.17)} \\ &\leq \xi_n^k(x) && \text{(by assumption)} \\ &\leq \xi_{n+1}(x + k) && \text{(Lemma 6.1.18),} \end{aligned}$$

a contradiction. \square

REMARK 6.1.23. The so-called *inverse Ackermann function*, often denoted by α , is not strictly speaking the inverse of the Ackermann function, but roughly speaking it grows as slowly as the Ackermann function grows quickly. It has many applications in the analysis of algorithms (for example for the disjoint-set data structure). One may assume that $\alpha(n) \leq 5$ for any practical input size n .

Exercises.

- (87) The *Fibonacci function* $f \in \mathcal{O}^{(1)}$ is defined via $f(0) := 0$, $f(1) := 1$, and $f(n+2) := f(n+1) + f(n)$ for all $n \in \mathbb{N}$. Show that f is primitive recursive.
- (88) Show that for every fixed n the operation ξ_n is primitive recursive.
- (89) (from Hils and Loeser [13]) The set of *elementary recursive functions* \mathcal{E} is the smallest clone that contains c_0^0 , addition $+$, multiplication \cdot , and the binary \div from Definition 6.1.3, and that is closed under *bounded sum* and *bounded product*: if $f \in \mathcal{E}^{(n+1)}$ then the operations

$$\begin{aligned} (x_1, \dots, x_n, x) &\mapsto \sum_{i=0}^x f(x_1, \dots, x_n, i) \\ (x_1, \dots, x_n, x) &\mapsto \prod_{i=0}^x f(x_1, \dots, x_n, i) \end{aligned}$$

are contained in $\mathcal{E}^{(n)}$. Show that the following operations are elementary recursive:

- for every $n, k \in \mathbb{N}$, the constant operation c_k^n ;

- $\lambda x. 2^x$.
 - $\lambda x, y. x^y$.
- (90) (from Hils and Loeser [13]) Let $t \in \mathcal{O}^{(2)}$ be the operation defined by $t(m, 0) := m$ and $t(m, n + 1) = 2^{t(m, n)}$.
- Prove that t is primitive recursive.
 - Prove that for every $n \in \mathbb{N}$ the function $t_n := \lambda x. t(x, n)$ is strictly increasing.
 - Prove that for every elementary recursive function f there exists $n \in \mathbb{N}$ such that f is dominated by t_n .
 - Prove that t is not elementary recursive.

6.2. Recursive Functions

The Ackermann function illustrates that there are functions that can be computed (in a still informal sense) but that are not primitive recursive; we need to strengthen our recursion mechanism to also capture such functions.

DEFINITION 6.2.1 (total μ -operator). Let $g: \mathbb{N}^{n+1} \rightarrow \mathbb{N}$ be such that for all $x_1, \dots, x_n \in \mathbb{N}$ there exists $y \in \mathbb{N}$ with $g(x_1, \dots, x_n, y) = 0$, then the function $\mu g: \mathbb{N}^n \rightarrow \mathbb{N}$ is defined as

$$(\mu g)(x_1, \dots, x_n) := \min\{y \in \mathbb{N} \mid g(x_1, \dots, x_n, y) = 0\}.$$

This definition has an important technical disadvantage: the definition only works for operations such that for all $x_1, \dots, x_n \in \mathbb{N}$ there exists $y \in \mathbb{N}$ with $g(x_1, \dots, x_n, y) = 0$. This restriction can become a problem later, because the condition is very difficult to check already if g is a primitive recursive function. We therefore work in the following with *partial operations*.

DEFINITION 6.2.2. A *partial operation of arity n over \mathbb{N}* is a function $f: A \rightarrow \mathbb{N}$ for some $A \subseteq \mathbb{N}^n$. We also write $\text{dom}(f)$ for A . If $\bar{t} \in \mathbb{N}^n \setminus A$, then we say that f is *undefined on \bar{t}* . A partial function is *total* if $\text{dom}(f) = \mathbb{N}^n$.

DEFINITION 6.2.3 (general μ -operator). Let g be a partial operation of arity $n + 1$ on \mathbb{N} . Then μg denotes the partial operation of arity n on \mathbb{N} that is defined as follows. If $x_1, \dots, x_n \in \mathbb{N}$ are such that there exists $y \in \mathbb{N}$ with $g(\bar{x}, y) = 0$ and $(\bar{x}, z) \in \text{dom}(g)$ for all $z \leq y$, then $\mu g(\bar{x})$ is defined to be the *minimal* such y ; otherwise, $\mu g(\bar{x})$ is undefined.

If we have a set of partial operations over \mathbb{N} , then closure under composition and closure under recursion are defined analogously to the case of total operations (Definition 6.1.1):

- if g is a partial operation of arity n over \mathbb{N} and f_1, \dots, f_n are partial operations of arity m over \mathbb{N} , then $g(f_1, \dots, f_n)$ is the partial operation of arity m over \mathbb{N} which maps $\bar{x} \in \mathbb{N}^m$ to $g(f_1(\bar{x}), \dots, f_n(\bar{x}))$ if $\bar{x} \in \text{dom}(f_i)$ for all i and $(f_1(\bar{x}), \dots, f_n(\bar{x})) \in \text{dom}(g)$, and is undefined otherwise.³
- if g is a partial operation of arity p over \mathbb{N} and h is a partial operation of arity $p + 2$ over \mathbb{N} , then $\rho(g, h)$ is the partial operation of arity $p + 1$ over \mathbb{N} defined by
 - $f(\bar{x}, 0) := g(\bar{x})$ if $\bar{x} \in \text{dom}(g)$, and otherwise f is not defined for $(\bar{x}, 0)$;

³This definition follows the usual definitions of composition of partial operations in partial clones; see, e.g., [5]. However, also see Exercise 91. One of the appealing features of our definition is the statement about composition that we prove in Exercise 93.

- $f(\bar{x}, y + 1) := h(\bar{x}, y, f(\bar{x}, y))$ if f is defined for (\bar{x}, y) (this property is defined simultaneously by induction on y) and if $(\bar{x}, y, f(\bar{x}, y)) \in \text{dom}(h)$, and otherwise f is not defined for $(\bar{x}, y + 1)$.

DEFINITION 6.2.4. The set of *partial recursive functions* is defined to be the smallest set of partial operations over \mathbb{N} that contains all primitive recursive functions and is closed under composition, recursion, and μ -recursion. A *(total) recursive function* is a total function which is partial recursive. We write \mathcal{R} for the clone of all recursive operations.

Church's thesis is the (non-mathematical) statement that every operation over \mathbb{N} which is computable in an intuitive sense is recursive.

Exercises.

- (91) Let $f := (\mu s(\pi_1^2))$. Determine $\pi_2^2(f, \pi_1^2)$. Note that the answer is *not* π_1^2 ! Also see Exercise 93.
- (92) (*) Show that the Ackermann function is recursive. If this exercise is too difficult, read on, it will become an easy exercise later (Exercise 98).

6.3. Recursively Enumerable Sets

DEFINITION 6.3.1. A set $X \subseteq \mathbb{N}^n$ is called

- *decidable* (or *recursive*) if 1_X is recursive;
- *recursively enumerable* (or *semi-decidable*) if there is a decidable set $Y \subseteq \mathbb{N}^{n+1}$ such that

$$X = \pi(Y) := \{(x_1, \dots, x_n) \mid (x_1, \dots, x_n, y) \in Y\}.$$

Every decidable set is recursively enumerable (Exercise 96), but the converse does not hold, as we will prove in Section 6.4.5 (Remark 6.4.25). The concept of recursively enumerable sets can be used to establish results about decidable sets, via the following theorem.

THEOREM 6.3.2 (Theorem of the Complement). *A set $X \subseteq \mathbb{N}^n$ is decidable if and only if both X and $\mathbb{N}^n \setminus X$ are recursively enumerable.*

PROOF. The forward implication is clear. For the converse, assume that $X \subseteq \mathbb{N}^n$ is such that there are decidable sets $Y, Y' \subseteq \mathbb{N}^{n+1}$ such that $X = \pi(Y)$ and $\mathbb{N}^n \setminus X = \pi(Y')$. Let $g := \lambda z_1, \dots, z_n, t ((1 \dot{-} 1_Y(\bar{z}, t)) \dot{-} 1_{Y'}(\bar{z}, t))$. Then $1_X(\bar{z}) = 1_Y(\bar{z}, (\mu g)(\bar{z}))$. \square

The set of recursively enumerable sets is a rather robust set of subsets of \mathbb{N} , as we will see in the following.

PROPOSITION 6.3.3. *Let $f_1, \dots, f_m \in \mathcal{R}^{(n)}$ and let $X \subseteq \mathbb{N}^m$ be recursively enumerable. Then*

$$X' := \{\bar{x} \in \mathbb{N}^n \mid (f_1(\bar{x}), \dots, f_m(\bar{x})) \in X\}$$

is recursively enumerable as well.

PROOF. There exists a decidable set $Y \subseteq \mathbb{N}^{m+1}$ such that $X = \pi(Y)$. Note that $Y' := \{(\bar{x}, y) \in \mathbb{N}^{n+1} \mid (f_1(\bar{x}), \dots, f_m(\bar{x}), y) \in Y\}$ is decidable. Then $X' = \pi(Y')$ and hence is recursively enumerable. \square

PROPOSITION 6.3.4. *The set of recursively enumerable sets is closed under projection, intersection, and union.*

PROOF. To show closure under projections, suppose that $n \geq 1$ and let $R \subseteq \mathbb{N}^n$ be recursively enumerable, so that there exists $S \subseteq \mathbb{N}^{n+1}$ such that $R = \pi(S)$. Note that

$$S' := \{(\bar{x}, t) \in \mathbb{N}^n \mid (\bar{x}, \beta_1^2(t), \beta_2^2(t)) \in S\}$$

is decidable (where β_i^2 is the function from Lemma 6.1.11). Thus, $\pi(R) = \pi(\pi(S)) = \pi(S')$ is recursively enumerable.

To show closure under intersection and union, let $R_1, R_2 \subseteq \mathbb{N}^n$ be recursively enumerable; so there are decidable sets $S_1, S_2 \subseteq \mathbb{N}^{n+1}$ such that $R_i = \pi(S_i)$ for $i \in \{1, 2\}$. Define

$$S'_i := \{(\bar{x}, y) \in \mathbb{N}^{n+1} \mid (\bar{x}, \beta_i^2(y)) \in S_i\}.$$

We then have

$$R_1 \cup R_2 = \pi(S_1 \cup S_2)$$

$$R_1 \cap R_2 = \pi(S'_1 \cap S'_2). \quad \square$$

EXAMPLE 29. Let $p(x_1, \dots, x_n, y_1, \dots, y_k)$ be a polynomial. Let X_p be the set $\{(a_1, \dots, a_k) \in \mathbb{N}^k \mid \text{there are } b_1, \dots, b_n \in \mathbb{N} \text{ such that } p(a_1, \dots, a_k, b_1, \dots, b_n) = 0\}$. It is now easy to see that the set X_p is recursively enumerable. However, there are polynomials p such that X_p is undecidable [21]; this will not be shown in this course. \triangle

The second item of the following theorem motivates the name *recursively enumerable*: the recursive functions f_1, \dots, f_n ‘enumerate’ the elements of the recursively enumerable set X . The third item, on the other hand, motivates the name *semi-decidable*.

THEOREM 6.3.5. *For $X \subseteq \mathbb{N}^n$ the following are equivalent.*

- (1) X is recursively enumerable.
- (2) $X = \emptyset$ or $X = \{(f_1(x), \dots, f_n(x)) \mid x \in \mathbb{N}\}$ where $f_1, \dots, f_n \in \mathcal{R}^{(1)}$.
- (3) $X = \text{dom}(g)$ for a partial recursive function g .
- (4) There exists a primitive recursive set $Y \subseteq \mathbb{N}^{n+1}$ such that $X = \pi(Y)$.

PROOF. Lemma 6.1.11 implies that it suffices to show the statement for $n = 1$.

The implication from (4) to (1) is trivial. For the implication from (1) to (2), let $Y \subseteq \mathbb{N}^2$ be decidable such that $X = \pi(Y)$. If $X = \emptyset$ then there is nothing to be shown; otherwise, let $r \in X$. Define $f: \mathbb{N} \rightarrow \mathbb{N}$ by

$$f(x) := \begin{cases} \beta_1^2(x) & \text{if } (\beta_1^2(x), \beta_2^2(x)) \in Y \\ r & \text{else.} \end{cases}$$

Clearly, $f \in \mathcal{R}^{(1)}$ and the image of f equals X .

For the implication from (2) to (3), if $X = \emptyset$ then $X = \text{dom}(\mu s(\pi_1^2))$. Otherwise, suppose that $X = \{f(x) \mid x \in \mathbb{N}\}$ for some $f \in \mathcal{R}^{(1)}$. Let

$$g := \mu(\lambda y, x. (f(x) \dot{\div} y) + (y \dot{\div} f(x))),$$

i.e., g returns the smallest x such that $f(x) = y$, if such an x exists, and is undefined otherwise. Then g is a partial recursive function such that $\text{dom}(g) = \text{im}(f)$.

The implication from (3) to (4) is at this point of the notes outside of our comfort zone. However, it will be an easy consequence of the results in the next section. The proof can be found at the end of Section 6.4.4 (Remark 6.4.22). \square

We are now in an awkward situation:

- We did not provide good arguments for Church’s thesis (end of Section 6.2).
- Exercise 92 was too difficult.

- In the proof of Theorem 6.3.5 there is an important gap.

All of this illustrates that we don't really understand the class of recursive functions yet. The next section will improve the situation a lot.

Exercises.

- (93) Show that if g is a partial recursive function of arity n over \mathbb{N} , and if f_1, \dots, f_n are partial recursive functions of arity m over \mathbb{N} , and the domains of g, f_1, \dots, f_n are recursive, then the domain of $g(f_1, \dots, f_n)$ is recursive as well.
- (94) Come up with other reasonable definitions of composition for partial operations where the answer to Exercise 91 is π_1^2 . Does your definition then also have the property from the previous exercise?
- (95) Show that decidable sets are closed under intersection and complement.
- (96) Show that every decidable set is recursively enumerable.

6.4. Turing Computable Functions

Turing machines were invented by Alan Turing in 1936 [31]. They are horrible to program. So the reader might wonder: why do we introduce Turing machines? Aren't there machine models that are more convenient for programming, after more than 80 years have passed since their discovery? My answer to the reader is: *yes*, there are such machine models, but for the purposes of this course they will probably lead to more work! The reason is that more advanced machine models require more work to simulate computation in these models. Recall from the beginning of this section: we need a machine model such that there is a machine in that model that can simulate any other machine in the same model. So we would like to keep the machine model simple.

On the other hand, there are many ways to simplify my definition of Turing machines even further (we will comment on a few simplifications later), and there are also completely different simplistic models of computation (see, e.g., Exercise 102). But then programming in such restricted models of computation is even more painful than it is with my definition of Turing machines.

To summarise: Turing machines strike a good balance between the following two opposite requirements that a theoretician has for a good machine model:

- the model should be simple, so that it can be simulated easily;
- the model should be powerful, so that it is easy to simulate other machines.

6.4.1. Turing machines. Our presentation is based on [13], but deviates in several details. A *Turing machine* is a tuple $M = (n, \ell, Q, s, t, \Sigma, \delta)$ consisting of

- an integer $n \geq 0$ (the number of *input tapes* of the machine),
- an integer $\ell \geq 0$ (the number of *work tapes* of the machine),
- a finite set Q (the *states* of the machine),
- $s, t \in Q$ (where s is called the *start state* and t is called the *terminal state*),
- the 3-element set $\Sigma := \{\$, |, \square\}$ (called the *alphabet*; $\$$ is the so-called *tape start symbol* and \square is the so-called *blank symbol*) and
- a *transition function*

$$\delta: Q \times \Sigma^{1+n+\ell} \rightarrow Q \times \Sigma^{1+n+\ell} \times \{-1, 0, 1\}$$

such that

- (1) $\delta(t, \bar{a}) = (t, \bar{a}, 0)$ and
- (2) for every $q \in Q$ there exists $m \in \{0, 1\}$ and $q' \in Q$ such that

$$\delta(q, \$, \dots, \$) = (q', \$, \dots, \$, m).$$

REMARK 6.4.1. The **idea** is that the machine has $1 + n + \ell$ tapes: an output tape, n input tapes, and ℓ work tapes. It has a read-write head which is at a position p above the tapes, can look at the $1 + n + \ell$ symbols at this position and modify these symbols; then it changes the position by $+1$, by -1 , or leaves it unchanged; moreover, it can change to another state. How this is done precisely is specified by δ . The extra conditions on δ express that when the machine enters the terminal state, it stays there forever, and that when the machine only reads start symbols, it cannot move further to the left and must not overwrite the start symbols.

To formally define how M operates, we need to introduce *words*. The set of finite sequences of elements of Σ is denoted by Σ^* ; the elements of Σ^* are also called *words over the alphabet Σ* . The *length* of a word w is the length of the sequence and denoted by $|w|$. There is a unique word of length 0 (also called the *empty word*), which is denoted by ϵ . If $w = (w_1, \dots, w_k)$ and $i \in \mathbb{N}$ with $i \geq 1$ then

$$w[i] := \begin{cases} w_i & \text{if } i \in \{1, \dots, k\} \\ \square & \text{otherwise.} \end{cases}$$

If $a \in \Sigma$, then $w[i \mapsto a]$ is the word of length $k := \max(i, |w|)$ defined as follows: for $j \leq k$, define

$$w[i \mapsto a][j] := \begin{cases} w[j] & \text{if } i \neq j \text{ and } j \in \{1, \dots, |w|\} \\ a & \text{if } i = j \\ \square & \text{otherwise.} \end{cases}$$

For the following definitions, we fix a Turing machine M .

DEFINITION 6.4.2 (Configuration). A *configuration of M* is a tuple

$$(q, w_0, \dots, w_{n+\ell}, p) \in Q \times (\Sigma^*)^{1+n+\ell} \times \{1, 2, \dots\}.$$

If w is the word that starts with $\$$ followed by the word $|\dots|$ of length exactly $x \in \mathbb{N}$, possibly followed by arbitrarily many blank symbols, then we say that w *represents x* . Note that we represent numbers ‘*in unary*’; this suffices for our purposes because we are not concerned with computational complexity or practical feasibility at this point.

DEFINITION 6.4.3 (Start Configuration). The configuration

$$C_0 = (s, \$, w_1, \dots, w_n, \underbrace{\$, \dots, \$}_{\ell \text{ times}}, 1)$$

where w_i , for $i \in \{1, \dots, n\}$, is the word that represents x_i , is called the *start configuration of M on input \bar{x}* .

DEFINITION 6.4.4 (Successor configuration). Let $C = (q, w_0, \dots, w_{n+\ell}, p)$ be a configuration of M and let $(q', a_0, \dots, a_{n+\ell}, m) := \delta(q, w_0[p], \dots, w_{n+\ell}[p])$. Note that by condition (2) on δ , we must have $p' := p + m \geq 1$. The successor configuration of C equals $(q', w_0[p \mapsto a_0], \dots, w_{n+\ell}[p \mapsto a_{n+\ell}], p')$.

DEFINITION 6.4.5 (Computation). A *computation of M* is a sequence C_0, \dots, C_r of configurations of M such that C_0 is the start configuration of M on some input \bar{x} and C_i is the successor configuration of C_{i-1} for every $i \in \{1, \dots, r\}$. The machine M *computes* a partial operation f of arity n over \mathbb{N} if for every $x_1, \dots, x_n \in \mathbb{N}$ we have that $(x_1, \dots, x_n) \in \text{dom}(f)$ if and only if there exists a computation C_0, \dots, C_r of M such that

$$C_r = (t, u_0, u_1, \dots, u_n, v_1, \dots, v_\ell, p)$$

for some $u_0, u_1, \dots, u_n, v_1, \dots, v_\ell \in \Sigma^*$ and $p \in \mathbb{N}$ such that

- u_0 represents $f(x_1, \dots, x_n)$,
- u_i represents x_i for every $i \in \{1, \dots, n\}$,
- v_i represents 0 for every $i \in \{1, \dots, \ell\}$.

If M computes a partial operation f of arity n over \mathbb{N} and $(x_1, \dots, x_n) \in \text{dom}(f)$, then we say that M halts on x_1, \dots, x_n .

LEMMA 6.4.6. *The constant function c_0^0 , the projections π_i^n , and the successor function s can be computed by a Turing machine.*

PROOF. The function c_0^0 can be computed by a Turing machine with a single tape, states $Q := \{s, t\}$, and a transition function δ such that $\delta(s, \$) = (t, \$, 0)$.

The function π_i^n can be computed by a Turing machine with $n + 1$ tapes, states $Q := \{s, t\}$, and a transition function δ such that

$$\delta(s, \$, \dots, \$) = (s, \$, \dots, \$, 1)$$

$$\delta(s, \square, a_1, \dots, a_n) = \begin{cases} (s, |, a_1, \dots, a_n, 1) & \text{if } a_i = | \\ (t, \square, a_1, \dots, a_n, 0) & \text{otherwise.} \end{cases}$$

The successor function on \mathbb{N} can be computed by a 2-tape Turing machine with states $Q := \{s, t\}$ and a transition function δ such that

$$\begin{aligned} \delta(s, \$, \$) &= (s, \$, \$, 1) \\ \delta(s, \square, |) &= (s, |, |, 1) && \text{(copy all |'s)} \\ \delta(s, \square, \square) &= (t, |, \square, 0) && \text{(add another |).} \end{aligned} \quad \square$$

REMARK 6.4.7. There are many variations of the definition of a Turing machine that lead to the same set of partial functions. For example, we may restrict the machines to a single tape, we may restrict Σ to $\{|\, \square\}$, or we may restrict the tape head to only move left or right, not staying in place. These restrictions also do not affect the length of the computations up to some polynomial factor. Only if we are interested in low time complexities, the differences become important: for example, the problem of deciding whether an input string is a palindrome (such as *1001*) can be solved in linear time on a two-tape Turing machine, but requires quadratic time on a one-tape Turing machine.

Exercises.

- (97) Show that the addition operation $+: \mathbb{N}^2 \rightarrow \mathbb{N}$ can be computed by a Turing machine.

6.4.2. Partial recursive functions are Turing computable. In this section we show that every partial recursive function is Turing computable (Theorem 6.4.11).

LEMMA 6.4.8. *The set of Turing computable partial operations over \mathbb{N} is closed under composition.*

PROOF. Let M_1, \dots, M_q , for $q \geq 1$, be Turing machines that compute the partial operations f_1, \dots, f_q of arity p over \mathbb{N} , and let M_0 be a Turing machine that computes a partial operation g of arity q over \mathbb{N} . We have to show that the operation $g(f_1, \dots, f_q)$ is Turing computable. Rename the states of M_0, M_1, \dots, M_q so that these sets are pairwise disjoint. Let Q be the union over all those state sets; we will define a Turing machine M with state set Q . We may assume that $s \in Q$ equals the start state of M_1 and that $t \in Q$ equals the terminal state of M_0 . If M_i has n_i tapes, for $i \in \{0, 1, \dots, q\}$, then M has

$$p + (n_0 - q) + \sum_{i=1}^q (n_i - p) \tag{8}$$

tapes. The idea is that M first executes the computation of M_1 using the states from M_1 , then executes the computation of M_2 using the states from M_2 , and so on, until M_q , all on separate work tapes (this accounts for the summand $\sum_{i=1}^q (n_i - p)$ in (8)). These machines write their output on the input tapes of M_0 (this accounts for the summand $n_0 - q$ in (8)). We may have to modify the machines so that after they have finished their computation they return to position 1. Finally, we execute the computation of M_0 on the output of the machines M_1, \dots, M_p . The resulting output will be the output of M . We leave the laborious details of the definition of the transition function for M to the reader. \square

LEMMA 6.4.9. *The set of Turing computable partial functions is closed under recursion.*

PROOF. Let M be a Turing machine with $1 + n + \ell$ tapes and a set of states Q computing a partial operation g of arity n over \mathbb{N} . Let M' be a Turing machine with $3 + n + \ell'$ tapes and a set of states Q' computing a partial operation h of arity $n + 2$ over \mathbb{N} . We have to construct a Turing machine N that computes the partial function $f := \rho(g, h)$ of arity $n + 1$ over \mathbb{N} . Our machine has $n + 4 + \ell + \ell'$ tapes, and its set of states is given by the disjoint union of Q and Q' and some additional states. During the entire computation, tape $n + 3$ will represent a natural number which is at most x_{n+1} ; this is true initially where tape $n + 3$ represents 0. For input $\bar{x} = (x_1, \dots, x_{n+1}) \in \mathbb{N}^{n+1}$ given on tapes $1, \dots, n, n + 1$, the machine proceeds as follows:

- (1) the machine computes $g(x_1, \dots, x_n)$ with input tapes $1, \dots, n$, output tape $n + 2$ and work tapes $n + 4, \dots, n + 4 + (\ell - 1)$, operating as M would do, up to renumbering the tapes.
- (2) Compare x_{n+1} and the natural number y which is represented on tape $n + 3$: if $x_{n+1} = y$, then go into the terminal state t .
- (3) Compute $h(\bar{x}, y, f(\bar{x}, y)) = f(\bar{x}, y + 1)$ by operating as M' would do, but on the input tapes $1, \dots, n, n + 3, n + 2$, output tape 0, and the final ℓ' tapes as auxiliary tapes.
- (4) Copy the content of tape 0 onto tape $n + 2$, then clear tape 0, and increment the content of tape $n + 3$ by one, that is, pass from y to $y + 1$. Then go back to step (2).

The final output $f(\bar{x})$ can then be found on tape $n + 2$. \square

It follows that every primitive recursive operation is Turing computable. Note that the program in Lemma 6.4.9 implements a ‘For $x = 1$ -to- n loop’ known to those who are familiar with imperative programming languages. The proof of the next lemma is simpler than the proof that we have just seen; instead of the ‘For-loop’ we have to implement a ‘While-loop’.

LEMMA 6.4.10. *The set of Turing computable partial functions is closed under μ -recursion.*

PROOF. Let M be a Turing machine with $n + \ell + 2$ tapes and a set of states Q computing a partial operation g of arity $n + 1$ over \mathbb{N} . We have to construct a Turing machine N that computes the partial function μg of arity n over \mathbb{N} . Our machine has $n + \ell + 2$ tapes, and its set of states is given by the states of Q and some additional states. For input $\bar{x} = (x_1, \dots, x_n) \in \mathbb{N}^n$ given on tapes $1, \dots, n$, the machine proceeds as follows:

- (1) the machine computes $g(x_1, \dots, x_n, y)$ for $y = 0$ as M would do.
- (2) If output tape 0 represents 0 then go into the terminal state t .

- (3) Increment y to $y + 1$ stored on output tape.
- (4) Return to tape position 1.
- (5) Go back to step (1).

The final output $\mu g(\bar{x})$ can then be found on tape $n + 1$. □

Note that the Turing machine in the proof of Lemma 6.4.10 might not terminate.

THEOREM 6.4.11. *Every partial recursive function is Turing computable.*

PROOF. By Lemma 6.4.6, the operations c_1^1 , s , and the projections are Turing computable. Lemma 6.4.8, Lemma 6.4.9, and Lemma 6.4.10 imply that the Turing-computable functions are closed under composition, recursion, and μ -recursion. Since the partial recursive functions are the smallest such class, the statement follows. □

The reader should by now be convinced by the (non-mathematical) fact that every function which is computable in an intuitive sense is also Turing computable. Quite surprisingly, we can establish the converse of Theorem 6.4.11 – so this will finally provide some evidence for Church’s thesis.

6.4.3. Turing computable functions are partial recursive. To show that every partial operation over \mathbb{N} that is Turing computable is also partial recursive, we first have to describe how to code a Turing machine as a natural number.

We identify the symbol \square with 0 the symbol $|$ with 1, and the symbol $\$$ with 2; moreover, we identify Q with $\{0, \dots, |Q| - 1\}$ where $s = 0$ and $t = 1$. A word $w \in \Sigma^*$ is coded as

$$\ulcorner w \urcorner := \sum_{i \in \{1, \dots, |w|\}} w[i] 3^i.$$

Let $M = (n, \ell, Q, s, t, \Sigma, \delta)$ be a Turing machine. To define a code for M , we use for fixed $k \in \mathbb{N}$ the operation $\alpha_k: \mathbb{N}^k \rightarrow \mathbb{N}$ introduced in Lemma 6.1.11.

- If $U = (q, s_0, \dots, s_{n+\ell}) \in Q \times \Sigma^{n+\ell+1}$, then

$$\ulcorner U \urcorner := \alpha_2(q, \ulcorner s_0 \dots s_{n+\ell} \urcorner).$$

- If $V = (q, s_0, \dots, s_{n+\ell}, m) \in Q \times \Sigma^{1+n+\ell} \times \{-1, 0, 1\}$, then

$$\ulcorner V \urcorner := \alpha_3(q, \ulcorner s_0 \dots s_{n+\ell} \urcorner, m+1).$$

- The transition function of M is then coded as

$$\ulcorner \delta \urcorner := \prod_{U \in Q \times \Sigma^{1+n+\ell}} \pi(\ulcorner U \urcorner) \ulcorner \delta(U) \urcorner$$

where $\pi: \mathbb{N} \rightarrow \mathbb{N}$ was introduced in Lemma 6.1.13.

- Finally, $\ulcorner M \urcorner := \alpha_4(n, \ell, |Q|, \ulcorner \delta \urcorner)$.

Note that if M and N are distinct Turing machines, then $\ulcorner M \urcorner \neq \ulcorner N \urcorner$. Also note that the transition function can be decoded from $\ulcorner M \urcorner$ in the following formal sense: there is a primitive recursive function that, on input $(\ulcorner M \urcorner, \ulcorner U \urcorner)$ where $U \in Q \times \Sigma^{1+n+\ell}$, computes $\ulcorner \delta(U) \urcorner$. This can be shown as in the proof of Lemma 6.1.15.

LEMMA 6.4.12. *The set*

$$\mathcal{M} := \{\ulcorner M \urcorner \mid M \text{ is a Turing machine}\}$$

is primitive recursive.

PROOF SKETCH. We use the primitive recursive operations from Lemma 6.1.11 and primitive recursive operations similar to the ones constructed in Lemma 6.1.15 to compute from a given $k \in \mathbb{N}$ a set of numbers that might correspond to a Turing machine M with $\ulcorner M \urcorner = k$. We have to check whether δ is indeed a function, etc.

The full details of this proof are lengthy to work out, but entirely straightforward and therefore omitted. \square

Not only Turing machines M , but also their computations can be coded. If $C = (q, w_0, \dots, w_{n+\ell}, p) \in Q \times (\Sigma^*)^{1+n+\ell} \times \mathbb{N}$ is a configuration of M , then

$$\ulcorner C \urcorner := \alpha_3(q, \langle \ulcorner w_0 \urcorner, \dots, \ulcorner w_{n+\ell} \urcorner \rangle, p).$$

where $\langle \cdot \rangle$ is defined in Definition 6.1.14.

The following lemma essentially states that from a given description of a Turing machine M and a configuration C of M , we can compute a description of the successor configuration of C .

LEMMA 6.4.13. *There is an operation $g_1 \in \mathcal{P}^{(2)}$ such that for all $m, c \in \mathbb{N}$:*

- *If $m \notin \mathcal{M}$, then $g_1(m, c) := 0$.*
- *Otherwise, $m = \ulcorner M \urcorner$ for some unique Turing machine M .*
 - *If there is no configuration C of M such that $c = \ulcorner C \urcorner$, then $g_1(m, c) := 0$.*
 - *Otherwise, $c = \ulcorner C \urcorner$ for some unique configuration C of M .*
Then $g_1(m, c) := \ulcorner C' \urcorner$ where C' is the successor configuration of C .

PROOF. Work. \square

The next lemma states that for a given description of a Turing machine M , an input (x_1, \dots, x_n) to the partial operation computed by M , and a number i , we can compute primitively recursively a description of the state of M when executed on input (x_1, \dots, x_n) for i many steps.

LEMMA 6.4.14. *There is an operation $g_2 \in \mathcal{P}^{(3)}$ such that for all $m, x, i \in \mathbb{N}$:*

- *If $m \notin \mathcal{M}$, then $g_2(m, x, i) := 0$.*
- *Otherwise, $m = \ulcorner M \urcorner$ for some unique Turing machine M with n input tapes.*
 - *If x is not of the form $\langle x_1, \dots, x_n \rangle$, then $g_2(m, x, i) := 0$.*
 - *Otherwise, g_2 returns $\ulcorner C' \urcorner$ for i -th successor configuration C' of the start configuration of M on input x_1, \dots, x_n .*

PROOF. Let g_5 be the operation of arity $2 + n$ that returns for given numbers $n, \ell, x_1, \dots, x_n \in \mathbb{N}$ the (unique) code for the starting configuration of any Turing machine M with n input tapes and ℓ work tapes where the input tapes represent x_1, \dots, x_n ; it is easy to see that g_5 is primitive recursive. The statement now follows using the operation g_1 from Lemma 6.4.13 and recursion. \square

The next lemma states that for a given description of a Turing machine M and an input to the partial operation computed by M we can compute recursively the smallest number $i \in \mathbb{N}$ where the machine executed on x reaches the terminal state t ; here we need the μ operator.

LEMMA 6.4.15. *There exists a binary partial recursive operation g_3 over \mathbb{N} such that for all $m, x \in \mathbb{N}$:*

- *If there is no $i \in \mathbb{N}$ such that $g_2(m, x, i)$ returns the code of a configuration of M in a **terminal** state, then $g_3(m, x)$ is undefined.*
- *Otherwise, $g_3(m, x)$ equals the smallest $i \in \mathbb{N}$ such that $g_2(m, x, i)$ returns the code of a configuration of M in a **terminal** state.*

PROOF. Clearly the set of Gödel numbers of configurations with an **terminal** state is decidable; the statement follows using μ -recursion and the primitive recursive function g_2 from Lemma 6.4.14. \square

The next theorem states that for a given description of a Turing machine M and an input to the partial operation computed by M , we can compute the output of M on that input.

THEOREM 6.4.16. *There exists a binary partial recursive operation g_4 over \mathbb{N} such that for all $m, x \in \mathbb{N}$:*

- *If $m \notin \mathcal{M}$, then $g_4(m, x)$ is undefined.*
- *Otherwise, $m = \ulcorner M \urcorner$ for some unique Turing machine M with n input tapes, computing a partial operation f of arity n over \mathbb{N} .*
 - *If x is not of the form $\langle x_1, \dots, x_n \rangle$, then $g_4(m, x)$ is undefined;*
 - *Otherwise, $g_4(m, x) := f(x_1, \dots, x_n)$.*

PROOF. It is straightforward to implement the case distinctions in the definition of the operation g_4 , using Lemma 6.4.12 and Lemma 6.1.15. Suppose now that $m = \ulcorner M \urcorner$ for some unique Turing machine with n input tapes that computes a partial operation f of arity n over \mathbb{N} and that $x = \langle x_1, \dots, x_n \rangle$ for some $x_1, \dots, x_n \in \mathbb{N}$.

Let g_2 be the operation from Lemma 6.4.14 and let g_3 be the operation from Lemma 6.4.15. Then

$$f(x_1, \dots, x_n) = g_4(m, x) := g_2(m, x, g_3(m, x)).$$

Moreover, g_4 is partial recursive since g_2 is primitive recursive and g_3 is partial recursive. \square

COROLLARY 6.4.17. *Every partial Turing computable function is recursive.*

PROOF. Let f be a partial operation of arity n over \mathbb{N} that is computed by a Turing machine M . Then for all $x_1, \dots, x_n \in \mathbb{N}$ we have

$$f(\bar{x}) = g_4(\ulcorner M \urcorner, \langle x_1, \dots, x_n \rangle)$$

where g_4 is the partial recursive function from Theorem 6.4.16. \square

COROLLARY 6.4.18. *A set $X \subseteq \mathbb{N}^n$ is decidable if and only if there exists a Turing machine that computes 1_X .*

PROOF. An immediate consequence of Corollary 6.4.17 and Theorem 6.4.11. \square

Exercises.

(98) Show that the Ackermann function is recursive.

6.4.4. Universal machines and universal functions. By combining the results of the previous two sections, we obtain a *universal machine*, i.e., a Turing machine U that for given $\ulcorner M \urcorner$ for some Turing machine M , simulates the computation of M on some given input \bar{x} , and returns the number that M would output on \bar{x} .

COROLLARY 6.4.19 (A universal machine). *There exists a 3-tape Turing machine U that computes the following binary partial operation u over \mathbb{N} . For $m, x \in \mathbb{N}$:*

- *If $m \notin \mathcal{M}$ then $u(m, x)$ is undefined.*
- *Otherwise, $m = \ulcorner M \urcorner$ for some unique Turing machine M that computes a partial operation f of arity n over \mathbb{N} . If x is not of the form $\langle x_1, \dots, x_n \rangle$, then $u(m, x)$ is undefined.*
- *Otherwise, $u(m, x) := f(x_1, \dots, x_n)$.*

PROOF. Let g_4 be the operation from Theorem 6.4.16. Theorem 6.4.11 implies that there exists a Turing machine U that computes g_4 . Then the statement follows from Theorem 6.4.16. \square

Similarly, we obtain a *universal partial recursive function*.

THEOREM 6.4.20. *Let $n \in \mathbb{N}$. Then there exists a partial recursive function u^n of arity $n + 1$ over \mathbb{N} such that for every partial recursive function f of arity n there exists $m \in \mathbb{N}$ such that $f = u_m^n := \lambda \bar{x}. u^n(m, \bar{x})$.*

PROOF. Let g_4 be the partial recursive operation from Theorem 6.4.16, and consider $u^n(m, x_1, \dots, x_n) := g_4(m, \langle x_1, \dots, x_n \rangle)$. If f is a partial recursive function, then there exists a Turing machine M that computes f by Theorem 6.4.11. Then we have $f(\bar{x}) = g_4(\ulcorner M \urcorner, \langle x_1, \dots, x_n \rangle)$. Since $\langle \cdot \rangle$ is primitive recursive, this implies the statement. \square

The proofs of Lemma 6.4.14 and Theorem 6.4.16 show as a by-product that if we have the total μ -operator, we don't need recursion in the definition of total recursive functions.

COROLLARY 6.4.21. *The set of total recursive functions is the smallest subset of \mathcal{O} which contains the primitive recursive operations and which is closed under composition and the total μ -operator.*

PROOF. Clearly, primitive recursive functions are total and recursive, and this set is closed under composition and the total μ -operator. Conversely, if f is total recursive, then by Theorem 6.4.11, there exists a Turing machine M that computes f . Since f is total, the operation defined by $x \mapsto g_3(\ulcorner M \urcorner, x)$ is total, and can be defined from the primitive recursive operation g_2 from Lemma 6.4.14 by the total μ -operator. Hence, for the operation g_4 from Theorem 6.4.16 we have that $x \mapsto g_4(\ulcorner M \urcorner, x)$ can be constructed from the primitive recursive operations using composition and the total μ -operator. Note that this operation equals f . \square

We can now first finish our proof of a statement about recursively enumerable sets that we have already announced in Section 6.3.

PROOF OF THEOREM 6.3.5, (3) \Rightarrow (4). Let g be a partial recursive function of arity 1 over \mathbb{N} . By Theorem 6.4.20 there exists $m \in \mathbb{N}$ with $g = u_m^1$. Hence,

$$X = \text{dom}(g) = \{y \in \mathbb{N} \mid \exists i. \beta_1^3(g_2(m, \langle y \rangle, i)) = 1\}$$

where g_2 is the primitive recursive operation from Lemma 6.4.14. **To see this, recall that $g_2(m, \langle y \rangle, i)$ is the code k of the configuration of the machine coded by m on input y after i steps, and $\beta_1^3(k) = 1$ if and only if this configuration is in the terminal state $t = 1$.** This proves that $X = \pi(Y)$ for some primitive recursive set Y . \square

REMARK 6.4.22. Note that the equivalence of (1) and (4) in Theorem 6.3.5 and the proof of (1) \Rightarrow (2) show that a subset of \mathbb{N}^n is recursively enumerable if and only if $X = \emptyset$ or $X = \{(f_1(x), \dots, f_n(x)) \mid x \in \mathbb{N}\}$ where f_1, \dots, f_n may even be chosen to be *primitive recursive*.

COROLLARY 6.4.23. *The set $S := \text{dom}(u^n) \subseteq \mathbb{N}^{n+1}$ is universal recursively enumerable in the sense that S is recursively enumerable and every recursively enumerable set $X \subseteq \mathbb{N}^n$ is of the form $\{\bar{x} \mid (m, \bar{x}) \in S\}$ for some $m \in \mathbb{N}$.*

Exercises.

(99) **Show that \mathcal{P} is not closed under the total μ -operator.**

6.4.5. The halting problem. Note that there are sets $X \subseteq \mathbb{N}$ that are not decidable: the reason is that there are uncountably many subsets of \mathbb{N} , but only countably many Turing machines. So by Cantor's theorem (Theorem 4.3.2) there exist subsets of \mathbb{N} that cannot be computed by Turing machines, and hence are not decidable by Corollary 6.4.18. In this section we present a *concrete* subset of \mathbb{N} that is not recursive; in fact, the subset we present will even be recursively enumerable. The proof is based on a *diagonal argument* like the proof of Theorem 4.3.2.

THEOREM 6.4.24. *Let $D := \text{dom}(\lambda x. u^1(x, x))$. Then $C := \mathbb{N} \setminus D$ is not recursively enumerable.*

PROOF. Suppose for contradiction that C is recursively enumerable; then $C = \text{dom}(u_i^1)$ for some $i \in \mathbb{N}$ by Corollary 6.4.23. Then $i \in C$ if and only if $u^1(i, i)$ is defined. On the other hand, $i \in D$ if and only if $u^1(i, i)$ is defined, a contradiction. \square

REMARK 6.4.25. Note that D is recursively enumerable, but not recursive, because otherwise $C := \mathbb{N} \setminus D$ would be recursive, and hence also recursively enumerable, contrary to Theorem 6.4.24.

We prove the following theorem by reducing to a situation where we can apply a similar idea as in the proof of Theorem 6.4.24 (but presented differently).

THEOREM 6.4.26 (Undecidability of the halting problem). *The set*

$$H_0 := \{m \in \mathcal{M} \mid m = \ulcorner M \urcorner \text{ and } M \text{ has no input tapes and halts}\}$$

is not recursive.

PROOF SKETCH. It suffices to show that the set

$$H := \{(m, x) \in \mathbb{N}^2 \mid m = \ulcorner M \urcorner \text{ and } M \text{ has 1 input tape and } M \text{ halts on } x\}$$

is not recursive (i.e., *undecidable*). The reason is that if M is a Turing machine with one input tape and $x \in \mathbb{N}$, then one can compute from $\ulcorner M \urcorner$ and x the number $\ulcorner N \urcorner$ for some Turing machine N such that M halts if and only if N halts on input x . **This is straightforward: essentially, M first writes the representation of x on its first work tape, and then simulates N in such a way that the first work tape of M plays the role of the input tape of N .**

Clearly, if H is decidable, then the *special halting problem*

$$H_s := \{\ulcorner M \urcorner \mid M \text{ has one input tape and } M \text{ halts on } \ulcorner M \urcorner\}$$

is decidable as well. So it suffices to show that H_s is undecidable. Suppose for contradiction that H_s is decidable, i.e., 1_{H_s} is recursive. By Theorem 6.4.11 there is a Turing machine M that computes 1_{H_s} . We construct a new Turing machine M' as follows:

- (1) Execute M on input w .
- (2) If M reaches a terminal state with 0 on the output tape, stop.
- (3) Otherwise: loop forever.

How does M' behave on input $\ulcorner M' \urcorner$?

$$\begin{aligned} M' \text{ halts on } \ulcorner M' \urcorner & \text{ iff } M \text{ computes 0 on input } \ulcorner M' \urcorner \\ & \text{ iff } \ulcorner M' \urcorner \notin H_s \\ & \text{ iff } M' \text{ executed on } \ulcorner M' \urcorner \text{ does not halt.} \end{aligned}$$

A contradiction! \square

THEOREM 6.4.27 (Rice's theorem). *Let \mathcal{X} be a set of unary partial recursive operations. Suppose that \mathcal{X} is neither empty nor equal to the set of all unary partial recursive functions over \mathbb{N} . Then*

$$I := \{\ulcorner M \urcorner \mid M \text{ computes a partial function in } \mathcal{X}\}$$

is undecidable.

PROOF SKETCH. Since I is decidable if and only if $\mathbb{N} \setminus I$ is decidable, we may assume that the function with the empty domain is not in \mathcal{X} , because otherwise we may pass to the complement. Choose $\ulcorner L \urcorner \in I$. Assume for contradiction that there is a Turing machine N that decides I . Then we create a new machine N' that decides the halting problem H_0 , which will be a contradiction to Theorem 6.4.26.

On input $\ulcorner M \urcorner$ for some Turing machine M with no input tapes,

- (1) N' computes $\ulcorner M' \urcorner$ where M' is the following Turing machine:
 - on input x , the machine M' first simulates the machine M .
 - Then M' simulates L on input x and returns the output of L .
- (2) Then N' simulates N on $\ulcorner M' \urcorner$.
- (3) If N accepts, then N' returns 1, otherwise N' returns 0.

We verify that N' indeed decides the halting problem:

$$\begin{aligned} N' \text{ returns 1 on input } \ulcorner M \urcorner & \text{ iff } N \text{ accepts } \ulcorner M' \urcorner \\ & \text{ iff } \ulcorner M' \urcorner \in I \\ & \text{ iff } M \text{ halts.} \end{aligned}$$

The last equivalence holds because if M halts, then M' computes the same function as L , and since $\ulcorner L \urcorner \in I$ we have $\ulcorner M' \urcorner \in I$. Conversely, if M does not halt, then M' computes the function with the empty domain. Since this function is by assumption not in \mathcal{X} , we conclude that $\ulcorner M' \urcorner \notin I$. \square

Exercises.

- (100) Show that the set of primitive recursive sets is not closed under projections.
- (101) Find a subset S of \mathbb{N} such that neither S nor its complement $\mathbb{N} \setminus S$ is recursively enumerable.
- (102) While Turing machines can be viewed as the origin of imperative programming languages, the following formalism, which goes back to Alonzo Church and Stephen Kleene in the 1930s, can be viewed as the origin of all functional programming languages. A λ -term is defined inductively as follows:
 - variables are λ -terms;
 - if M is a λ -term and x is a variable, then so is $\lambda x. M$ (*Abstraction*);
 - if M and N are λ -terms, then so is (MN) (*Application*).

If x is a variable and M, N are λ -terms, then the λ -term $M[x := N]$ obtained from M by substituting x with N is defined inductively as follows:

- if M is of the form x , then $M[x \mapsto N] := N$;
- if M is of the form $\lambda y. M'$ and x and y are distinct variable symbols, then

$$M[x \mapsto N] := \lambda y(M[x \mapsto N]);$$

- if M is of the form $\lambda x. M'$ then

$$M[x \mapsto N] := (\lambda x. M')$$

(all occurrences of x in M are bound, 'no substitution');

- if M is of the form $(M_1 M_2)$ then

$$M[x \mapsto N] := (M_1[x \mapsto N] M_2[x \mapsto N]).$$

A β -reduction step replaces a λ -term of the form $((\lambda x.M)N)$ by the λ -term $M[x \mapsto N]$. We then also write $((\lambda x.M)N) \xrightarrow{\beta} M[x \mapsto N]$. For example,

$$(\lambda x.x)(\lambda x.x) \xrightarrow{\beta} x[x \mapsto (\lambda x.x)] = \lambda x.x$$

Note that for example in the term $\lambda x.x$ no further β -reduction steps can be applied. Show that there are λ -terms such that repeatedly performing β -reduction does not terminate.

- (103) Show that the set of pairs (s, t) of λ -terms, coded as a set of natural numbers, such that t can be reached from s by repeated application of β -reduction steps is recursively enumerable.
- (104) (*) Show that the set from the previous exercise is not recursive.
- (105) (*) Let τ be a signature that only contains unary relation symbols.
- Show that τ -sentences have the *finite model property*: if a τ -sentence ϕ has a model, then it also has a finite model.
 - Derive that there exists an algorithm that tests whether a given τ -sentence is satisfiable.
- (106) Show the existence of a *quine*, which is a Turing machine M with no input tapes which writes $\ulcorner M \urcorner$ on its output tape.
- (107) **Is the following problem decidable: given (a syntactic description of) a primitive recursive function $f: \mathbb{N} \rightarrow \mathbb{N}$, is $f = c_0^1$?**

The Incompleteness Theorems

In this chapter we prove Gödel's first incompleteness theorem: every consistent and recursive theory that contains the axioms of *weak Peano arithmetic* is necessarily incomplete (Theorem 7.5.1). The same holds for ZF instead of weak Peano arithmetic.

We then also sketch a proof of Gödel's second incompleteness theorem: every consistent and recursive theory that contains *Peano arithmetic* cannot prove its own consistency (Theorem 7.7.11). It follows that if ZF is consistent, then there is no formal proof in ZF proving the consistency of ZF.

7.1. Coding Formulas and Proofs

We first describe how to code formulas into numbers, similarly as we have coded machines and computations into numbers in Section 6.4.3. Let $\tau = \{z_1, \dots, z_n\}$ be a finite signature; for each $i \in \{1, \dots, n\}$ the symbol z_i is either a relation or a function symbol. The set of *logical symbols* is the union of τ , a countably infinite set of variables v_1, v_2, \dots , and the set of symbols containing $=, \wedge, \neg, \exists, (, \text{ and })$. We assign to each τ -formula ϕ a *Gödel number* $\ulcorner \phi \urcorner \in \mathbb{N}$ as follows and start by assigning Gödel numbers to the logical symbols ($\langle \cdot \rangle$ has been defined in Lemma 6.1.15).

$$\frac{z \quad | \quad = \quad \wedge \quad \neg \quad \exists \quad (\quad) \quad z_i \quad v_i}{\ulcorner z \urcorner \quad | \quad \langle 0, 0 \rangle \quad \langle 0, 1 \rangle \quad \langle 0, 2 \rangle \quad \langle 0, 3 \rangle \quad \langle 0, 4 \rangle \quad \langle 0, 5 \rangle \quad \langle 0, i + 5 \rangle \quad \langle 1, i \rangle}$$

The Gödel number $\ulcorner w \urcorner$ of a word $w = w_1 \dots w_n \in \Sigma^*$ is defined as

$$\langle \ulcorner w_1 \urcorner, \dots, \ulcorner w_n \urcorner \rangle.$$

The proofs of the facts in Lemma 7.1.1, Lemma 7.1.2, and Theorem 7.1.3 can be seen as programming exercises.

LEMMA 7.1.1. *The following sets are primitive recursive.*

- $Term(\tau) := \{\ulcorner t \urcorner \mid t \text{ is } \tau\text{-term}\}.$
- $Formula(\tau) := \{\ulcorner \phi \urcorner \mid \phi \text{ is } \tau\text{-formula}\}.$
- $Sentence(\tau) := \{\ulcorner \phi \urcorner \mid \phi \text{ is } \tau\text{-sentence}\}.$

LEMMA 7.1.2. *There exists a primitive recursive function $tsubst \in \mathcal{P}^3$ such that for every $n \in \mathbb{N}$ and all τ -terms s, t*

$$tsubst(n, \ulcorner s \urcorner, \ulcorner t \urcorner) = \ulcorner t[v_n \mapsto s] \urcorner.$$

There exists a primitive recursive function $fsubst \in \mathcal{P}^3$ such that for every $n \in \mathbb{N}$, every τ -term s , and every τ -formula ϕ

$$fsubst(n, \ulcorner s \urcorner, \ulcorner \phi \urcorner) = \ulcorner \phi[v_n \mapsto s] \urcorner.$$

THEOREM 7.1.3. *The set $\{\ulcorner \phi \urcorner \mid \phi \text{ is a logical axiom for } \tau\}$ is primitive recursive.*

For a theory T , define

$$\ulcorner T \urcorner := \{\ulcorner \phi \urcorner \mid \phi \in T\}$$

$$\text{Prf}(T) := \{\langle \ulcorner \phi_1 \urcorner, \dots, \ulcorner \phi_n \urcorner \rangle \mid (\phi_1, \dots, \phi_n) \text{ is a formal proof in } T\}.$$

LEMMA 7.1.4. *Let T be a theory. If $\ulcorner T \urcorner$ is recursive, then $\text{Prf}(T)$ is recursive, and if $\ulcorner T \urcorner$ is primitive recursive, then $\text{Prf}(T)$ is primitive recursive. In particular, $\text{Prf}(\emptyset)$ is primitive recursive.*

PROPOSITION 7.1.5. *The set $\{\ulcorner \phi \urcorner \mid \vdash \phi\}$ is recursively enumerable.*

PROOF. We have $\vdash \phi$ if and only if there exist $n \in \mathbb{N}$ and first-order sentences ϕ_1, \dots, ϕ_n such that $(\phi_1, \dots, \phi_n) \in \text{Prf}(\emptyset)$ and $\phi_n = \phi$. \square

7.2. Decidable Theories

Let T be a τ -theory. The *deductive closure* of T is the set

$$\text{Thm}(T) := \{\phi \text{ a } \tau\text{-sentence} \mid T \vdash \phi\}.$$

A theory is called *finitely axiomatisable* if there exists a finite theory T' such that $\text{Thm}(T) = \text{Thm}(T')$.

EXAMPLE 30. The theory $\text{Th}(\mathbb{Q}; <)$ is finitely axiomatisable. Clearly, $(\mathbb{Q}; <)$ is an unbounded and dense linear order. Note that the fact that $<$ is an unbounded and dense linear order can be expressed by a first-order sentence ψ . Conversely, if ϕ is a first-order sentence that holds in $(\mathbb{Q}; <)$, then it must hold in all countable dense linear orders: this follows from the well-known and easy fact that all countable dense linear orders without endpoints are isomorphic (this was observed already by Cantor [6]). In fact, it can also be shown that ϕ must hold in *all* countable dense linear orders (we refer to a model theory course for more details) and hence $\text{Th}(\mathbb{Q}; <) = \text{Thm}(\{\psi\})$. \triangle

EXAMPLE 31. The theory $\text{Th}(\mathbb{N}; =)$ is not finitely axiomatisable. \triangle

Finite axiomatisability is quite rare, so we need a more general concept. We say that T is *recursively* (or *effectively*) *axiomatisable* if there exists a τ -theory T' such that $\text{Thm}(T) = \text{Thm}(T')$ and $\ulcorner T' \urcorner$ is recursive.

EXAMPLE 32. Every finitely axiomatisable theory is effectively axiomatisable. \triangle

PROPOSITION 7.2.1. *A theory T is effectively axiomatisable if and only if $\ulcorner \text{Thm}(T) \urcorner$ is recursively enumerable.*

PROOF. First suppose that there exists a theory T' such that $\text{Thm}(T) = \text{Thm}(T')$ and $\ulcorner T' \urcorner$ is recursive. We have $\phi \in \text{Thm}(T)$ if and only if there exists a formal proof of ϕ in T' . The statement follows because $\text{Prf}(T')$ is recursive by Lemma 7.1.4.

Conversely, suppose that $\ulcorner \text{Thm}(T) \urcorner$ is recursively enumerable. By Theorem 6.3.5 (1) \Rightarrow (2) there exists a recursive function $f \in \mathcal{R}^{(1)}$ such that $\ulcorner \text{Thm}(T) \urcorner = \{f(i) \mid i \in \mathbb{N}\}$. Let ϕ_i be such that $\ulcorner \phi_i \urcorner = f(i)$. The function $g: \mathbb{N} \rightarrow \mathbb{N}$ given by

$$g(n) := \ulcorner \bigwedge_{i \in \{0, \dots, n\}} \phi_i \urcorner$$

is recursive and we have that $g(n) \geq n$ for every $n \in \mathbb{N}$. The image of g is therefore a recursive set. Since

$$\text{Thm}\left(\bigwedge_{i \in \{0, \dots, n\}} \phi_i \mid n \in \mathbb{N}\right) = \text{Thm}(T)$$

this proves that T is effectively axiomatisable. \square

DEFINITION 7.2.2. A theory T is called

- *recursive* if $\ulcorner T \urcorner$ is recursive.
- *decidable* if $\ulcorner \text{Thm}(T) \urcorner$ is recursive.

THEOREM 7.2.3. *Let T be a complete τ -theory which is effectively axiomatisable. Then T is decidable.*

PROOF. By Proposition 7.2.1 we know that $\ulcorner \text{Thm}(T) \urcorner$ is recursively enumerable. Its complement $\mathbb{N} \setminus \ulcorner \text{Thm}(T) \urcorner$ equals

$$\{\ulcorner \phi \urcorner \mid \neg \phi \in \text{Thm}(T)\} \cup \{a \mid a \notin \text{Sentence}(\tau)\}.$$

The first set in this union is recursively enumerable since $\ulcorner \phi \urcorner \mapsto \ulcorner \neg \phi \urcorner$ is given by a (primitive) recursive function. The second set is primitive recursive by Lemma 7.1.1. Hence, the union is recursively enumerable as well by Proposition 6.3.4. Theorem 6.3.2 therefore implies that $\ulcorner \text{Thm}(T) \urcorner$ is recursive. \square

EXAMPLE 33. The theory $T := \text{Th}(\mathbb{Q}; <)$ is finitely axiomatisable (Example 30), and hence it is effectively axiomatisable (Example 32). Of course, T is complete and thus Theorem 7.2.3 implies that T is decidable. \triangle

Exercises.

- (108) (Exercise 5.2.6 in Hils and Loeser [13]) Let T be a τ -theory. If T is decidable and S is a finite set of first-order τ -sentences, then $T \cup S$ is decidable.
 (109) Is $\text{Th}(\mathbb{N}; \neq)$ finitely axiomatisable? Is it recursive? Is it decidable?

7.3. (Weak) Peano Arithmetic

In this section we work with the signature $\tau_{\text{Arithm}} := \{0, s, +, \cdot, <\}$ of arithmetic. For every $n \in \mathbb{N}$ we define the term \underline{n} by induction, setting $\underline{0} := 0$ and $\underline{n+1} := s(\underline{n})$.

DEFINITION 7.3.1. The set PA_0 of *weak Peano axioms*, also known as *Robinson's theory Q*, is the first-order theory consisting of the following eight sentences:

- | | | |
|----|---|---------------------------------------|
| A1 | $\forall v_0. s(v_0) \neq 0$ | (0 is not a successor) |
| A2 | $\forall v_0 \exists v_1 (v_0 \neq 0 \Rightarrow s(v_1) = v_0)$ | (non-zero elements have predecessor) |
| A3 | $\forall v_0, v_1 (s(v_0) = s(v_1) \Rightarrow v_0 = v_1)$ | (injectivity of s) |
| A4 | $\forall v_0 (v_0 + 0 = v_0)$ | (0 is additively neutral) |
| A5 | $\forall v_0, v_1 (v_0 + s(v_1) = s(v_0 + v_1))$ | (distributivity of $+$ over s) |
| A6 | $\forall v_0 (v_0 \cdot 0 = 0)$ | (0 is multiplicative zero) |
| A7 | $\forall v_0, v_1 (v_0 \cdot s(v_1) = (v_0 \cdot v_1) + v_0)$ | (distributivity of \cdot over s) |
| A8 | $\forall v_0, v_1 (v_0 < v_1 \Leftrightarrow \exists v_2 (v_1 = v_0 + s(v_2)))$ | (definition of $<$) |

Note that we could alternatively work with the signature without $<$ and the first seven axioms only, because $<$ is first-order definable from the other symbols in the signature (Axiom 8 provides such a definition).

At first sight, weak Peano arithmetic appears to be very weak: there are models of PA_0 in which multiplication is not commutative and where there are elements x such that $s(x) = x$, as the following example illustrates. But we will see later that even in such a weak theory we can express surprisingly strong things with first-order sentences (Theorem 7.3.8).

EXAMPLE 34. Consider the structure \underline{A} whose domain A is $\mathbb{N} \cup \{a\}$ where a is some new element, not contained in \mathbb{N} . Define

$$\begin{aligned} 0^{\underline{A}} &:= 0 \\ s^{\underline{A}}(x) &:= \begin{cases} x+1 & \text{if } x \in \mathbb{N}, \\ a & \text{if } x = a \end{cases} \\ x +^{\underline{A}} y &:= \begin{cases} x+y & \text{if } x, y \in \mathbb{N}, \\ a & \text{otherwise} \end{cases} \\ x \cdot^{\underline{A}} y &:= \begin{cases} xy & \text{if } x, y \in \mathbb{N}, \\ 0 & \text{if } y = 0, \\ a & \text{otherwise.} \end{cases} \end{aligned}$$

This finishes our description of \underline{A} because $0^{\underline{A}}$ and $+^{\underline{A}}$ uniquely determine $<^{\underline{A}}$. Note that $s^{\underline{A}}(a) = a$ and that $x \cdot^{\underline{A}} 0 = 0 \neq a = 0 \cdot^{\underline{A}} a$. Note that $\underline{A} \models \text{PA}_0$. \triangle

Exercises.

(110) Does PA_0 have finite models?

DEFINITION 7.3.2. The set PA of *Peano axioms* consists of PA_0 together with the set of all first-order sentences of the form

$$\forall v_1, \dots, v_n ((\phi(0, \bar{v}) \wedge \forall v_0 (\phi(v_0, \bar{v}) \Rightarrow \phi(s(v_0), \bar{v}))) \Rightarrow \forall v_0. \phi(v_0, \bar{v}))$$

for a first-order τ_{Arithm} -formula $\phi(v_0, v_1, \dots, v_n)$.

Clearly, the structure $\underline{N}_{\text{st}} := (\mathbb{N}; 0, s, +, \cdot, <)$ with the usual definition of the successor function, addition, multiplication, and the order, is a model of PA . Note that the natural numbers, the successor function, addition, multiplication, and the order were all formalised in ZF in Chapter 4; it follows that if ZF is consistent, then so is PA . If \underline{A} is a model of PA , then an element $a \in A$ is called *non-standard* if $a \neq \underline{n}^{\underline{A}}$ for every $n \in \mathbb{N}$. By compactness of first-order logic, there are models of PA with non-standard elements.

LEMMA 7.3.3. For every $n \in \mathbb{N}$

$$\text{PA}_0 \models \forall x (x < \underline{n} \Leftrightarrow \bigvee_{i=0}^{n-1} x = \underline{i}).$$

PROOF. Let $\underline{A} \models \text{PA}_0$. We show the statement by induction on n . For $n = 0$, suppose for contradiction that $\underline{A} \models c < 0$ for some $c \in A$. Then by A8 there exists an $a \in A$ such that $\underline{A} \models a + c = 0 \wedge c \neq 0$. Hence $c = s(d)$ for some d by A2, and

$$\begin{aligned} 0 &= a + s(d) \\ &= s(a + d) \end{aligned} \quad (\text{by A5})$$

in contradiction to A1. For the induction step, we first prove that

$$\text{PA}_0 \models \forall x, y (x < y \Leftrightarrow s(x) < s(y)). \quad (9)$$

This follows from A8, using that if $a, b, c \in A$ then

$$\begin{aligned} \underline{A} \models c + a = b \wedge a \neq b &\Leftrightarrow \underline{A} \models s(c + a) = s(b) \wedge s(a) \neq s(b) && (\text{by A3}) \\ &\Leftrightarrow \underline{A} \models s(c) + a = s(b) \wedge s(a) \neq s(b) && (\text{by A5}). \end{aligned}$$

Now let $a \in A$. We have to show that $a < \underline{n+1}^A$ if and only if $a = \underline{i}$ for some $i < n+1$. The statement holds for $a = 0$ because A8 implies that $0 < c$ for any $c \neq 0$. Otherwise, there exists $b \in A$ such that $s(b) = a$ by A2. Then

$$\begin{aligned} a < \underline{n+1}^A &\Leftrightarrow b < \underline{n}^A && \text{(by (9))} \\ &\Leftrightarrow b = \underline{i}^A \text{ for some } i < n && \text{(by inductive assumption)} \\ &\Leftrightarrow a = \underline{i}^A \text{ for some } i < n+1 && \text{(by (9)). } \quad \square \end{aligned}$$

LEMMA 7.3.4. *Let \underline{A} be a model of PA_0 . Then $N := \{\underline{n}^A \mid n \in \mathbb{N}\}$ is the domain of a substructure of \underline{A} which is isomorphic to \underline{N}_{st} .*

PROOF. Let $i: \mathbb{N} \rightarrow N$ be the map $n \mapsto \underline{n}^A$. One verifies easily by (naive) induction on $n \in \mathbb{N}$ that i preserves 0 , s , $+$, and \cdot . For example, for any $m, n \in \mathbb{N}$ one has $\text{PA}_0 \models \underline{m} + \underline{n} = \underline{m+n}$, using (A4) and (A5). The injectivity of i follows from Lemma 7.3.3 (Exercise 112). It follows from (A8) that i preserves $<$. The verification that i also preserves the complement of $<$ is left as an exercise. \square

DEFINITION 7.3.5. Let $f \in \mathcal{O}^{(k)}$. A τ_{Arithm} -formula $\phi(x_0, x_1, \dots, x_k)$ *represents* f if for all $n_1, \dots, n_k \in \mathbb{N}$ one has

$$\text{PA}_0 \models \forall y (\phi(y, \underline{n}_1, \dots, \underline{n}_k) \Leftrightarrow y = \underline{f(n_1, \dots, n_k)}).$$

Let $R \subseteq \mathbb{N}^k$. A τ_{Arithm} -formula $\phi(x_1, \dots, x_k)$ *represents* R if for all $n_1, \dots, n_k \in \mathbb{N}$ one has

$$\begin{aligned} (n_1, \dots, n_k) \in R &\text{ then } \text{PA}_0 \models \phi(\underline{n}_1, \dots, \underline{n}_k) \\ \text{if } (n_1, \dots, n_k) \notin R &\text{ then } \text{PA}_0 \models \neg \phi(\underline{n}_1, \dots, \underline{n}_k). \end{aligned}$$

DEFINITION 7.3.6. The set of Σ_1 -formulas is the smallest set of τ_{Arithm} -formulas that contains all quantifier-free formulas and which is stable under existential quantification $\exists x$, conjunction \wedge , disjunction \vee , bounded universal quantification of the form $\forall x (x < t \Rightarrow \phi)$ where t is a term not depending on the variable x . We also write $\forall x < t. \phi$ instead of $\forall x (x < t \Rightarrow \phi)$.

The set of *strict* Σ_1 -formulas is the smallest set of τ_{Arithm} -formulas containing the formulas $0 = x$, $s(x) = y$, $x + y = z$, $x \cdot y = z$, $x = y$, $\neg(x = y)$, $x < y$, $\neg(x < y)$ and which is stable under conjunction, disjunction, existential quantification, and universal quantification of the form $\forall x < y. \phi$, assuming that x and y are two distinct variables.

LEMMA 7.3.7. *Any Σ_1 -formula is equivalent to a strict Σ_1 -formula.*

PROOF. One may eliminate complex terms by using existential quantifiers. \square

For example, $s(x) + y = z$ is equivalent to

$$\exists x' (s(x) = x' \wedge x' + y = z).$$

THEOREM 7.3.8. *Every Σ_1 -sentence that holds in \underline{N}_{st} is a consequence of PA_0 .*

PROOF. By Lemma 7.3.7, it suffices to prove the statement for strict Σ_1 -sentences. We prove that for any strict Σ_1 -formula $\phi(x_1, \dots, x_n)$ and $m_1, \dots, m_n \in \mathbb{N}$ we have $\underline{N}_{st} \models \phi(m_1, \dots, m_n) \Rightarrow \text{PA}_0 \models \phi(\underline{m}_1, \dots, \underline{m}_n)$. We prove this by structural induction over the definition of formulas, starting from the case when ϕ is an atomic formula or a negated atomic formula which follows from Lemma 7.3.4.

Assume that the statement holds for $\phi(x_0, x_1, \dots, x_n)$ and for $\psi(x_0, x_1, \dots, x_n)$. Clearly it then also holds for $\phi \wedge \psi$ and $\phi \vee \psi$. To prove that it holds for $\exists x_0. \phi$, suppose that $\underline{N}_{st} \models (\exists x_0. \phi)(m_1, \dots, m_n)$. Then there exists $m_0 \in \mathbb{N}$ such that

$\underline{N}_{\text{st}} \models \phi(m_0, m_1, \dots, m_n)$, which by the inductive assumption implies that $\text{PA}_0 \models \phi(\underline{m}_0, \underline{m}_1, \dots, \underline{m}_n)$. Hence, $\text{PA}_0 \models (\exists x_0. \phi)(\underline{m}_1, \dots, \underline{m}_n)$.

Finally, suppose that $\underline{N}_{\text{st}} \models (\forall x_0 < x_1. \phi)(\underline{m}_1, \dots, \underline{m}_n)$. Then by definition of bounded universal quantification one has $\underline{N}_{\text{st}} \models \phi(m_0, m_1, \dots, m_n)$ for every $m_0 < m_1$, and by inductive assumption $\text{PA}_0 \models \phi(\underline{m}_0, \underline{m}_1, \dots, \underline{m}_n)$ for every $m_0 < m_1$. In other words, $\text{PA}_0 \models \bigvee_{i=0}^{m_1-1} \phi(i, \underline{m}_1, \dots, \underline{m}_n)$. Lemma 7.3.3 then implies that $\text{PA}_0 \models \forall x_0 < \underline{m}_1. \phi(x_0, \underline{m}_1, \dots, \underline{m}_n)$. \square

THEOREM 7.3.9 (Representability theorem). *Every recursive function can be represented by a Σ_1 -formula.*

(BEGINNING OF PROOF). We already know from Corollary 6.4.21 that every recursive function can be constructed from primitive recursive operations by composition and the total μ -operator. It is clear that the successor function s , the constant function c_0^0 , and the projections are representable. By the trick from Lemma 7.3.7 we may also represent compositions of representable functions. Suppose now that f has the representation $\phi(z, \bar{y}, x)$. Then the following formula $\psi(z, \bar{y})$ is a presentation of μf :

$$\phi(0, \bar{y}, z) \wedge \forall z' < z (\neg \phi(0, \bar{y}, z')).$$

We are left with the task to verify that the class of representable functions is also closed under recursion (cf. Definition 6.2.4); this is the mentioned complication. However, it turns out that recursion is not needed in the definition of recursive functions! The reason is that recursion can be simulated by total μ -recursion; proving this requires some efforts, though. \square

As a temporary definition, let \mathcal{R}_0 be the smallest clone that contains s , c_0^0 , and is closed under total μ -recursion. To prove that $\mathcal{R}_0 = \mathcal{R}$ (Lemma 7.3.12) we basically follow Section 6.1, replacing recursion by total μ -recursion. Note that this does not follow from Corollary 6.4.21 because we do not require in the definition of \mathcal{R}_0 that it contains \mathcal{P} . Subsets of \mathbb{N}^n whose indicator function is in \mathcal{R}_0 are called 0-recursive (temporarily, since we are about to prove that 0-recursive sets and recursive sets are the same thing).

LEMMA 7.3.10. $\mathcal{R}_0^{(3)}$ contains \div (see Lemma 6.1.4). The set of 0-recursive sets is closed under Boolean combinations, bounded quantification, and contains

$$\{(x, y, z) \in \mathbb{N}^3 \mid x \equiv y \pmod{z}\}.$$

Moreover, the set \mathcal{R}_0 is stable under definition by cases (see Lemma 6.1.8).

The final ingredient to the proof of Theorem 7.3.9 is an ingenious idea of Gödel.

LEMMA 7.3.11 (Gödel's β -function). *There exists an operation $\beta \in \mathcal{R}_0^{(3)}$ such that for any finite sequence (c_0, \dots, c_{n-1}) of natural numbers there are $a, b \in \mathbb{N}$ with $\beta(a, b, i) = c_i$ for $i \in \{0, \dots, n-1\}$.*

PROOF SKETCH. Define

$$\beta(a, b, i) := \min\{z \in \mathbb{N} \mid z \equiv a \pmod{(i+1)b+1}\}.$$

It is easy to write this definition using the total μ -operator using functions whose containment in \mathcal{R}_0 we already know. To prove that β has the required properties, let c_0, \dots, c_{n-1} be given and let $b \in \mathbb{N}$ be such that b is divisible by $n!$ and $b > c_i$ for all i . Then $b+1, 2b+1, 3b+1, \dots, nb+1$ is a sequence of pairwise prime integers. By the Chinese Remainder Theorem there exists $a \in \mathbb{N}$ such that

$$a \equiv c_i \pmod{(i+1)b+1}$$

for all $i \in \{0, \dots, n-1\}$. As $c_i < (i+1)b+1$ for all i , it is the smallest natural number which is congruent to a modulo $((i+1)b+1)$. \square

LEMMA 7.3.12 (Elimination of recursion). $\mathcal{R} = \mathcal{R}_0$.

PROOF. By Corollary 6.4.21 it suffices to show that \mathcal{R}_0 is closed under recursion, because then by definition \mathcal{R}_0 contains the primitive recursive operations, and hence equals \mathcal{R} by Corollary 6.4.21. Let $g \in \mathcal{R}_0^{(n)}$ and $h \in \mathcal{R}_0^{(n+2)}$ and let $f \in \mathcal{O}^{(n+1)}$ be the operation obtained from g and h by recursion. Consider the set

$$Z := \{(\bar{x}, y, a, b) \in \mathbb{N}^{n+3} \mid \beta(a, b, 0) = g(\bar{x}) \text{ and for all } i \leq y \\ \beta(a, b, i+1) = h(\bar{x}, i, \beta(a, b, i))\}$$

whose indicator function is in \mathcal{R}_0 by Lemma 7.3.11. Moreover, the property of β implies that for any $(\bar{x}, y) \in \mathbb{N}^{n+1}$ there are $a, b \in \mathbb{N}$ such that $(\bar{x}, y, a, b) \in Z$. Thus, $e := \lambda \bar{x}, y. \min\{s \mid \exists a, b \leq s. (\bar{x}, y, a, b) \in Z\}$ is in \mathcal{R}_0 . Finally, $f(\bar{x}, y)$ equals

$$\min\{z \mid \exists a, b \leq e(\bar{x}, y). (\bar{x}, y, a, b) \in Z \text{ and } z = \beta(a, b, y)\}$$

proving that $f \in \mathcal{R}_0$. \square

This completes the proof of Theorem 7.3.9.

COROLLARY 7.3.13. Every recursively enumerable set can be represented by a Σ_1 -formula.

PROOF. Let $R \subseteq \mathbb{N}^n$ be of the form $\pi(Y)$ for some recursive set Y . By Theorem 7.3.9 there exists a Σ_1 -formula ϕ that represents 1_Y . Then $\exists y. \phi(0, x_1, \dots, x_n, y)$ represents R . \square

Exercises.

- (111) Prove that the structure \underline{A} from Example 34 does not satisfy PA.
- (112) Prove that the map i in the proof of Lemma 7.3.4 is injective.
- (113) Prove that the map i in the proof of Lemma 7.3.4 also preserves the complement of $<$.
- (114) A relation $R \subseteq \mathbb{N}^n$ is called *arithmetical* if it is first-order definable¹ in the structure $\underline{N}_{\text{st}} = (\mathbb{N}; 0, s, +, \cdot, <)$. Show that every recursively enumerable set is arithmetical.
- (115) Show that the theory $\text{Th}(\underline{N}_{\text{st}})$ is undecidable (Definition 7.2.2).
- (116) Is $\text{Th}(\underline{N}_{\text{st}})$ finitely axiomatisable?
- (117) Show that there are finite signatures such that the empty theory over this signature is undecidable. Show that the empty theory over the empty signature is decidable.
- (118) Is PA a complete theory? Try to answer this question without using results that come later in the course notes.
- (119) Show that

$$\text{PA} \vdash \forall x, y. x + y = y + x,$$

indicating carefully which axioms of Peano arithmetic are used, and where.

- (120) Do the ordinal numbers with respect to $<$ form a model of WPA? Is this a properly phrased question?

¹First-order definability was introduced in Section 3.2.4.

7.4. The Theorems of Tarski and Church

We start this section with a proof of the fixed point theorem in logic, which states that for every first-order definable property E of the natural numbers there exists a first-order sentence ψ in the language of arithmetic which states that

“My Gödel number has property E .”

We prove this with a diagonalisation argument, similar to the one used in the proof of Cantor’s theorem (Theorem 4.3.2), the existence of recursively enumerable sets that are not recursive (Theorem 6.4.24), and the undecidability of the halting problem (Theorem 6.4.26).

Lemma 7.1.2 implies that there exists a primitive recursive function subst such that if $\phi(x)$ is a τ_{Arithm} -formula and $n \in \mathbb{N}$, then $\text{subst}(\ulcorner \phi \urcorner, n) = \ulcorner \phi(\underline{n}) \urcorner$. By the representability theorem (Theorem 7.3.9), there exists a Σ_1 -formula $G(z, x, y)$ representing subst . For a given formula $\phi(x)$, define

$$\begin{aligned} H_\phi(x) &:= \exists z (G(z, x, x) \wedge \phi(z)) \\ n_\phi &:= \ulcorner \neg H_\phi \urcorner \\ \Delta_\phi &:= \neg H_\phi(n_\phi). \end{aligned}$$

PROPOSITION 7.4.1 (The diagonal argument). *Let $\phi(x)$ be a τ_{Arithm} -formula. Then*

$$\text{PA}_0 \vdash (\phi(\ulcorner \Delta_\phi \urcorner) \Leftrightarrow \neg \Delta_\phi).$$

PROOF. Note that

$$\text{subst}(n_\phi, n_\phi) = \text{subst}(\ulcorner \neg H_\phi \urcorner, n_\phi) = \ulcorner \Delta_\phi \urcorner. \quad (10)$$

Let \underline{A} be a model of PA_0 . Then

$$\begin{aligned} \underline{A} \models \phi(\ulcorner \Delta_\phi \urcorner) &\Leftrightarrow \underline{A} \models \phi(\text{subst}(n_\phi, n_\phi)) && \text{(by (10))} \\ &\Leftrightarrow \underline{A} \models \exists z (G(z, \underline{n}_\phi, \underline{n}_\phi) \wedge \phi(z)) && \text{(by the definition of } G) \\ &\Leftrightarrow \underline{A} \models H_\phi(\underline{n}_\phi) && \text{(by the definition of } H_\phi) \\ &\Leftrightarrow \underline{A} \models \neg \Delta_\phi && \text{(by the definition of } \Delta_\phi). \quad \square \end{aligned}$$

COROLLARY 7.4.2 (Fixed point theorem). *For any τ_{Arithm} -formula $\phi(x)$ there exists a τ_{Arithm} -sentence ψ such that*

$$\text{PA}_0 \vdash (\phi(\ulcorner \psi \urcorner) \Leftrightarrow \psi).$$

PROOF. This follows from Theorem 7.4.1 by taking $\psi := \Delta_{\neg\phi}$:

$$\text{PA}_0 \vdash (\neg\phi(\ulcorner \Delta_{\neg\phi} \urcorner) \Leftrightarrow \neg\Delta_{\neg\phi}) \quad \square$$

THEOREM 7.4.3 (Tarski’s theorem on the Non-definability of Truth). *Let $\underline{A} \models \text{PA}_0$. Then there is no τ_{Arithm} -formula $\phi(x)$ such that for every τ_{Arithm} -sentence ψ one has $\underline{A} \models \psi$ if and only if $\underline{A} \models \phi(\ulcorner \psi \urcorner)$.*

PROOF. Suppose for contradiction that $\phi(x)$ is a τ_{Arithm} -sentence as in the statement of the theorem. By Proposition 7.4.1 we have $\text{PA}_0 \vdash \phi(\ulcorner \Delta_\phi \urcorner) \Leftrightarrow \neg \Delta_\phi$. Hence, $\underline{A} \models \phi(\ulcorner \Delta_\phi \urcorner)$ if and only if $\underline{A} \models \neg \Delta_\phi$. However, by assumption $\underline{A} \models \phi(\ulcorner \Delta_\phi \urcorner)$ if and only if $\underline{A} \models \Delta_\phi$, a contradiction. \square

The following is much stronger than the statement from Exercise 115.

THEOREM 7.4.4 (Church’s theorem). *Let T be a consistent τ_{Arithm} -theory that contains PA_0 . Then T is undecidable.*

PROOF. Suppose for contradiction that $\ulcorner \text{Thm}(T) \urcorner$ would be recursive. Then by the representability theorem (Theorem 7.3.9) there exists a Σ_1 -formula $\phi(x)$ representing $\ulcorner \text{Thm}(T) \urcorner$. Applying Proposition 7.4.1 we would get

$$\begin{aligned} T \vdash \Delta_\phi &\Leftrightarrow \ulcorner \Delta_\phi \urcorner \in \ulcorner \text{Thm}(T) \urcorner && \text{(by the definition of } \ulcorner \cdot \urcorner \text{ and Thm)} \\ &\Leftrightarrow \text{PA}_0 \vdash \phi(\ulcorner \Delta_\phi \urcorner) && \text{(by the assumptions on } \phi) \\ &\Leftrightarrow \text{PA}_0 \vdash \neg \Delta_\phi && \text{(by Proposition 7.4.1)} \end{aligned}$$

This is a contradiction: if $\text{PA}_0 \vdash \phi(\ulcorner \Delta_\phi \urcorner)$, then by the above we have $T \vdash \Delta_\phi$ and $\text{PA}_0 \vdash \neg \Delta_\phi$, which implies $T \vdash \neg \Delta_\phi$ since T contains PA_0 , and hence T is inconsistent, a contradiction. If $\text{PA}_0 \not\vdash \phi(\ulcorner \Delta_\phi \urcorner)$, then $\text{PA}_0 \vdash \neg \phi(\ulcorner \Delta_\phi \urcorner)$ since ϕ represents a subset of \mathbb{N} ; we then obtain a contradiction similarly as above. \square

COROLLARY 7.4.5 (Undecidability of first-order logic). *There exists a finite signature τ such that*

$$\{\ulcorner \phi \urcorner \mid \phi \text{ is valid } \tau\text{-sentence}\}$$

is undecidable.

7.5. Gödel's First Incompleteness Theorem

The following is Gödel's first incompleteness theorem [11] in an elegant strengthening that was found by Rosser [25].

THEOREM 7.5.1 (Gödel-Rosser). *Let T be a consistent and recursive τ_{Arithm} -theory that contains PA_0 . Then T is incomplete.*

PROOF. If T were complete, then T would be decidable by Theorem 7.2.3, contradicting Church's theorem (Theorem 7.4.4). \square

Note that every model of ZF contains in particular \mathbb{N} as an element, and that the relations $0, s, +, \cdot, <$ on \mathbb{N} are first-order definable in ZF, so the following should not come as a surprise.

COROLLARY 7.5.2. *Let T be a recursive and consistent $\{\in\}$ -theory that contains ZF. Then T is incomplete.*

PROOF SKETCH. Construct a consistent and recursive $\{\in\} \cup \tau_{\text{Arithm}}$ -theory T' that contains T and PA_0 , and apply Theorem 7.5.1. \square

Exercises.

- (121) Find a Turing machine M so that the question whether M halts, formalised in ZF, is independent from ZF.
- (122) Can you modify your solution to Exercise 118 to prove Theorem 7.5.1?

7.6. Σ_1 -Definability of Truth of Σ_1 -Sentences

The formula True_{Σ_1} introduced in Proposition 7.6.1 below plays an important role in the formulation of Gödel's second incompleteness theorem. Starting from here, everything that follows is bonus material for the course, included for the interested reader.

PROPOSITION 7.6.1. *There exists a Σ_1 -formula $\text{True}_{\Sigma_1}(x)$ such that for every Σ_1 -sentence ϕ*

$$\text{PA} \vdash (\phi \Leftrightarrow \text{True}_{\Sigma_1}(\ulcorner \phi \urcorner)).$$

We mention that here it is essential that we use Peano arithmetic PA rather than weak Peano arithmetic PA₀; also note that the assumption that ϕ is a Σ_1 -sentence is necessary, because for general first-order formulas the statement would be false by Tarski's theorem on the non-definability of truth (Theorem 7.4.3).

7.7. Gödel's Second Incompleteness Theorem

Informally, Gödel's second incompleteness theorem states that the consistency of PA cannot be shown in PA (and similarly, that the consistency of ZF cannot be shown within ZF). In order to phrase this statement formally, we must clarify what we mean by 'showing the consistency of PA within PA'. Similarly as in [13], we follow the presentation of Martin Ziegler [32].

Throughout this section, let T be a recursive τ_{Arithm} -theory that contains PA. To formulate the second incompleteness theorem of Gödel we have to construct a specific Σ_1 -formula that expresses that a sentence is provable in T . We use the following sentences in the construction.

- $\text{Ax}_T(x)$ for the Σ_1 -formula that represents the set of all Gödel numbers of logical axioms and of sentences in T ,
- $\text{MP}(x, y, z)$ for the Σ_1 -formula that represents the set of triples $(\ulcorner \phi \urcorner, \ulcorner \psi \urcorner, \ulcorner \delta \urcorner)$ with ϕ, ψ, δ such that δ is obtained from ϕ and ψ by modus ponens.

It is clear that such formulas exist by the representability theorem (Theorem 7.3.9). In our construction, we will also use the primitive recursive component function $(x)_i$ from Lemma 6.1.15: the representability theorem implies that there is a Σ_1 -formula $\phi(x_1, x_2, x_3)$ that represents this function, and hence we may freely use it in terms of Σ_1 -formulas.

DEFINITION 7.7.1. Let $\text{Proof}_T(n)$ be the formula

$$\exists a \forall i \leq n (\text{Ax}_T((a)_i) \vee \exists j, k < i. \text{MP}((a)_j, (a)_k, (a)_i)) \quad (11)$$

$$\vee \text{True}_{\Sigma_1}((a)_i). \quad (12)$$

We write $\Box_T \phi$ for $\text{Proof}_T(\ulcorner \phi \urcorner)$.

If we dropped in the formula $\text{Proof}_T(n)$ the disjunct in (12), then the correspondingly modified sentence $\Box_T \phi$ would express exactly that ϕ has a formal proof in T . The addition of the disjunct in (12) is motivated by the following proposition, which fails otherwise (if PA is consistent; see Exercise 124).

PROPOSITION 7.7.2. *Let ϕ be a Σ_1 -sentence. Then*

$$\text{PA} \vdash (\phi \Rightarrow \Box_T \phi).$$

PROOF. If $\text{PA} \vdash \phi$ then $\text{PA} \vdash \text{True}_{\Sigma_1}(\ulcorner \phi \urcorner)$ and hence $\text{PA} \vdash \Box_T \phi$. \square

The following properties (L1)-(L3) are called *Loeb's axioms*, or sometimes the *Hilbert-Bernays provability conditions*.

THEOREM 7.7.3. *Let ϕ and ψ be first-order τ_{Arithm} -sentences. Then \Box_T satisfies the following properties:*

- | | | |
|------|--|--------------------------|
| (L1) | $\text{If } T \vdash \phi \text{ then } T \vdash \Box_T \phi$ | (necessitation) |
| (L2) | $T \vdash (\Box_T \phi \wedge \Box_T (\phi \rightarrow \psi)) \Rightarrow \Box_T \psi$ | (internal modus ponens) |
| (L3) | $T \vdash \Box_T \phi \Rightarrow \Box_T \Box_T \phi$ | (internal necessitation) |

PROOF. (L1). If $T \vdash \phi$, then $\underline{N}_{\text{st}} \models \Box_T \phi$ (also see Exercise 124 (1)). Since $\Box_T \phi$ is a Σ_1 -sentence, it holds in any model of PA₀ by Theorem 7.3.8, so in particular in every model of T .

(L2): Let \underline{A} be a model of $T \cup \{\Box_T \phi, \Box_T(\phi \Rightarrow \psi)\}$. Then there exists an element in A that represents a formal proof of ϕ , and an element that represents a formal proof of $\phi \Rightarrow \psi$, and since $\underline{A} \models \text{PA}$ we may find an element that represents a formal proof of ψ by combining the formal proofs above and applying modus ponens. So $\underline{A} \models \Box_T \psi$; hence, the statement follows from the completeness theorem.

(L3): This is a special case of Proposition 7.7.2 since $\Box_T \phi$ is a Σ_1 -sentence. \square

COROLLARY 7.7.4. *Let ϕ and ψ be τ_{Arithm} -sentences.*

$$\text{If } T \vdash \phi \Rightarrow \psi, \text{ then } T \vdash \Box_T \phi \Rightarrow \Box_T \psi$$

PROOF. Assume $T \vdash \phi \Rightarrow \psi$. Then $T \vdash \Box_T(\phi \Rightarrow \psi)$ by (L1), and hence (L2) implies that $T \vdash \Box_T \phi \Rightarrow \Box_T \psi$. \square

REMARK 7.7.5. The operator \Box_T provides an abstract view on provability so that we may forget about the details of the definition of $\Box_T \phi$ as a τ_{Arithm} -sentence; this idea goes back to Solovay [29]. A good example that illustrates this is Corollary 7.7.4, where we have only used Loeb's axioms. This perspective is formalised in *provability logic*, which in our case leads to the *modal logic K*; see e.g. [4] for a full treatment. For example, (L1) is known in modal logic under the name **N** and (L3) is known under the name **4**. Provability logic can be used to analyse other logics as well. In our case, the only fact that we still need to establish before we can fully work in the framework of provability logic is Theorem 7.7.6.

THEOREM 7.7.6 (Modal fixed points). *Let ϕ be a τ_{Arithm} -sentence. Then there exists a τ_{Arithm} -sentence ψ such that*

$$T \vdash \psi \Leftrightarrow (\Box_T \psi \Rightarrow \phi).$$

PROOF. There exists a τ_{Arithm} -formula $\theta(x)$ such that for any τ_{Arithm} -sentence η

$$T \vdash \theta(\ulcorner \eta \urcorner) \Leftrightarrow (\Box_T \eta \Rightarrow \phi).$$

This follows from the observation that decoding Gödel numbers is primitive recursive and Theorem 7.3.9. We apply Corollary 7.4.2 to the formula θ and obtain

$$T \vdash \psi \Leftrightarrow \theta(\ulcorner \psi \urcorner)$$

which implies the statement. \square

From now on, it suffices to work with Loeb's axioms and the modal fixed point theorem. This is why we drop the reference to T and the signature τ_{Arithm} from now on until the statement of Gödel's second incompleteness theorem (they do not play any role in the arguments that follow).

THEOREM 7.7.7 (Loeb's theorem). *Let ϕ be a first-order sentence. Then*

$$\vdash \Box(\Box \phi \Rightarrow \phi) \Rightarrow \Box \phi.$$

PROOF. By Theorem 7.7.6, there exists a first-order sentence ψ such that

$$\vdash \psi \Leftrightarrow (\Box \psi \Rightarrow \phi). \tag{13}$$

Hence,

$$\vdash \Box\psi \Rightarrow \Box(\Box\psi \Rightarrow \phi) \quad (\text{by (13) and Corollary 7.7.4 (1)}) \quad (14)$$

$$\vdash \Box\psi \Rightarrow (\Box\Box\psi \Rightarrow \Box\phi) \quad (\text{by (14) and box distributivity}) \quad (15)$$

$$\vdash \Box\psi \Rightarrow \Box\phi \quad (\text{by (15) and internal necessitation}) \quad (16)$$

$$\vdash (\Box\phi \Rightarrow \phi) \Rightarrow (\Box\psi \Rightarrow \phi) \quad (\text{by (16)}) \quad (17)$$

$$\vdash (\Box\phi \Rightarrow \phi) \Rightarrow \psi \quad (\text{by (17) and (13)}) \quad (18)$$

$$\vdash \Box(\Box\phi \Rightarrow \phi) \Rightarrow \Box\psi \quad (\text{by (18) and (Corollary 13)}) \quad (19)$$

$$\vdash \Box(\Box\phi \Rightarrow \phi) \Rightarrow \Box\phi \quad (\text{by (19) and (16)}) \quad (20)$$

which concludes the proof. \square

REMARK 7.7.8. If we interpret $\Box\phi$ as

‘I believe ϕ ’

then Loeb’s theorem corresponds to *modesty*²: you do not believe that your belief in ϕ would imply that ϕ is true, without first believing that ϕ is true.

COROLLARY 7.7.9. *Let ϕ be a first-order sentence. If $\vdash \Box\phi \Rightarrow \phi$, then $\vdash \Box\phi$.*

PROOF. If $\vdash \Box\phi \Rightarrow \phi$ then $\vdash \Box(\Box\phi \Rightarrow \phi)$ by (L1), so the statement follows from Theorem 7.7.7. \square

COROLLARY 7.7.10. *Let ϕ be a first-order sentence. If $\vdash \neg\Box\phi$, then $\vdash \Box\phi$.*

Let $\text{Con}_T := \neg\Box_T(0 = 1)$, expressing the consistency of the τ_{Arithm} -theory T .

THEOREM 7.7.11 (Gödel’s Second Incompleteness Theorem). *Let T be a recursive τ_{Arithm} -theory that contains PA. If T is consistent, then $T \not\vdash \text{Con}_T$.*

PROOF. Suppose $T \vdash \text{Con}_T$. Then Corollary 7.7.10 implies that $T \vdash \Box_T(0 = 1)$, so T is inconsistent. \square

Again, we obtain the corresponding statement for ZF as a consequence.

COROLLARY 7.7.12. *Let T be a recursive and consistent $\{\in\}$ -theory that contains ZF. Then $T \not\vdash \text{Con}(T)$.*

Exercises.

(123) Prove *box distributivity*³:

$$\vdash \Box(\phi \Rightarrow \psi) \Rightarrow (\Box\phi \Rightarrow \Box\psi).$$

(124) Let $\text{Proof}'_T(n)$ be the formula $\text{Proof}_T(n)$ but without the disjunct in (12), and let $\Box'_T\phi$ be defined with $\text{Proof}'_T(n)$ instead of $\text{Proof}_T(n)$. Prove that

(a) for every τ_{Arithm} -sentence ϕ

$$T \vdash \phi \text{ if and only if } \underline{N}_{\text{st}} \models \Box'_T\phi.$$

(b) if ϕ is a Σ_1 -sentence, then

$$\underline{N}_{\text{st}} \models \phi \Rightarrow \underline{N}_{\text{st}} \models \Box'_T\phi.$$

(c) Show that \Box'_T satisfies (L1) and (L2).

(d) (*) Assuming that PA is consistent, show that Proposition 7.7.2 fails for $\Box'_T(n)$ instead of $\Box_T(n)$. In other words, there is a recursive τ_{Arithm} -theory T that contains PA and a Σ_1 -sentence ϕ such that

$$\text{PA} \not\vdash (\phi \Rightarrow \Box'_T\phi).$$

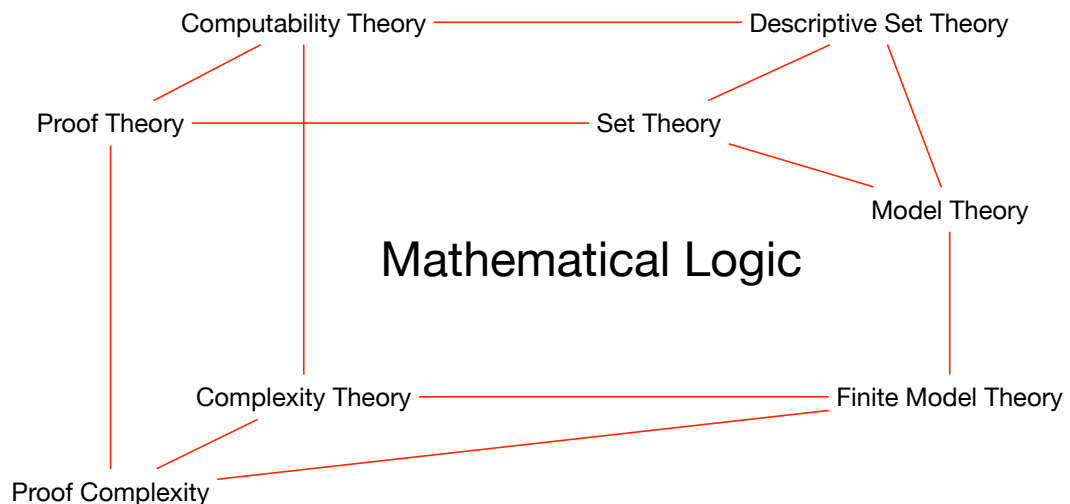
²This interpretation of the operator \Box gives rise to *doxastic logic*.

³In modal logic, box distributivity has the name **K**, not to be confused with K , which is $\{\mathbf{K}, \mathbf{N}\}$.

Further Reading

There are several avenues to continue from here. I would like to mention *model theory*, *proof theory*, *recursion theory* (also known as *computability theory*), *complexity theory*, *proof complexity*, *finite model theory*, *set theory*, and *descriptive set theory*.

- *Model theory* is concerned with *tame* first-order theories, that is, theories that are well-behaved so that we can at least partially understand how the models of the theory look like. Model theory can then be applied in many areas of mathematics. I recommend [15] and its long version [14] and [30]. The compactness theorem plays a central role in model theory, while the completeness theorem does not. Model theory can help a lot if we want to prove that certain first-order theories are decidable.
- *Complexity theory* is concerned with the power of computation if we impose resource restrictions. Computation is still formalised with Turing machines; typical resources are then *space* and *time*: we may impose bounds on the running time or the space consumption of the Turing machine, depending on the size of the input. A more subtle interesting resource is *alternation*; the concept of *non-deterministic computation* being a special case. Complexity theory has plenty of open problems. Many of these open problems belong to the most difficult problems in all of mathematics, the most famous open problem being the P=NP problem. Here, P is the class of all problems that can be solved by a Turing machine in polynomial time, and NP is the class of all problems that can be solved by a non-deterministic Turing machine in polynomial time. We recommend the text books [1, 23].



- *Proof complexity* is an area where the questions are motivated by complexity theory, but they are treated from the perspective of (propositional) proof theory. In this course we have seen an important propositional proof system, namely resolution. We have also hinted at a propositional proof system based on Modus Ponens, called *Frege's propositional calculus*. Many of the central questions in proof complexity are concerned with the size of shortest proofs in proof systems such as resolution and Frege. Similarly as complexity theory, this area is full of extremely difficult longstanding open problems. One of the most famous open problems is whether there are polynomial-size Frege proofs for all propositional tautologies, which corresponds to the question whether $\text{NP} = \text{coNP}$ in complexity theory. We recommend Krajíček's book which is freely available as a pdf document [19].
- One of the starting points of *finite model theory* is the observation that the class NP, unlike the class P, has a simple logical characterisation which does not involve Turing machines, namely *Fagin's theorem*: it states that a class \mathcal{C} of finite structures which is closed under isomorphism is in NP if and only if there exists a *second-order sentence* which holds on a finite structure if and only if the structure is in \mathcal{C} . Many other complexity classes can be characterised by a logic, in particular if we assume that the structure is ordered (such an order is implicitly given if we are interested in classes of words over some given finite alphabet). The questions in finite model theory are often motivated by complexity theory, but the methods often come from model theory; we recommend the text-books [7, 16, 20].
- *Proof theory*: in this first course we have seen a Hilbert-style proof system that allowed for a relatively short proof of the completeness theorem; however, we have also seen (mostly in the exercises) that already for very simple mathematical arguments it can be very difficult to translate them into our formalism. This motivates the search for proof systems where it is easier to come up with formal proofs. Such proof systems can be combined with automatic theorem provers so that tedious verification steps can be done by a computer. The correctness of these systems themselves can be formally verified as well! Such systems already exist, e.g., there is the system *Isabelle* developed by the University of Cambridge and TU München, or the system *Coq* developed by INRIA, CNRS, Ecole Polytechnique, Université Paris Diderot and others. The long-term vision is that mathematicians work interactively with such systems to eventually transform their proofs into formal proofs that can be checked efficiently by a program. A classic text book in proof theory is [26].
- *Set theory* is a deep area. One of the topics are *large cardinal properties*. Whether large cardinals exist cannot be proved in ZFC, and postulating the existence of such cardinals is a fruitful way to extend ZFC. One of the goals is to obtain a systematic understanding of the resulting extensions of ZFC. There are important connections with model theory and proof theory; I refer to the text-books [17, 22].
- *Recursion theory (computability theory)*: what can be computed by a Turing machine if the machine has access to an oracle that solves the halting problem, or the complement of the halting problem? Of course, there are again function from $\mathbb{N} \rightarrow \mathbb{N}$ that cannot be computed in this way, simply by a counting argument. In fact, there is a complicated landscape of larger and larger sets of functions over the natural numbers; the beauty of the subject is that, unlike in complexity theory, we have proofs showing that these

classes are indeed different. Recursion theory is quite old [12]; recent developments include for instance *continuous computability theory*, *Kolmogorov complexity*, and *reverse mathematics*.

- *Descriptive set theory* is the study of certain classes of well-behaved subsets of \mathbb{R} , the set of real numbers. Exercises 59 and 60 in this course are basic facts from descriptive set theory. The subject exists in a classical version [18] and an effective version [9] which creates a link with recursion theory. Descriptive set theory has applications e.g. in functional analysis and the study of group actions. Basic facts of descriptive set theory appear in the master course on automorphism groups at TU Dresden [3].

Bibliography

- [1] S. Arora and B. Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, 594 pages.
- [2] M. Bodirsky. Model theory, 2022. Course Notes, TU Dresden, <https://wwwpub.zih.tu-dresden.de/~bodirsky/Model-theory.pdf>.
- [3] M. Bodirsky. Automorphism groups, 2023. Course Notes, TU Dresden, <https://wwwpub.zih.tu-dresden.de/~bodirsky/Automorphism-Groups.pdf>.
- [4] G. Boolos. *The Logic of Provability*. Cambridge University Press, 1993.
- [5] F. Börner, L. Haddad, and R. Pöschel. Minimal partial clones. *Bulletin of the Australian Mathematical Society*, 44(3):405–415, 1991.
- [6] G. Cantor. Über unendliche, lineare Punktmannigfaltigkeiten. *Mathematische Annalen*, 23:453–488, 1884.
- [7] H.-D. Ebbinghaus and J. Flum. *Finite Model Theory*. Springer, Berlin, Heidelberg, New York, 1999. Second edition.
- [8] A. Fraenkel. Zu den Grundlagen der Cantor-Zermeloschen Mengenlehre (wesentlich Wiedergabe eines Vortrags a. d. deutschen Mathematikertag, Jena 921, vgl. Jahresb. d. D. Math.-Ver., 30 [1921], 97 f.). *Math. Annalen*, 86:230–237, 1922.
- [9] S. Gao. *Invariant Descriptive Set Theory*. Pure and applied mathematics. Taylor and Francis, 2008.
- [10] K. Gödel. Die Vollständigkeit der Axiome des logischen Funktionenkalküls. *Monatshefte für Mathematik (in German)*, 37(1):349–360, 1930.
- [11] K. Gödel. Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme, I. *Monatshefte für Mathematik und Physik*, 38(1):173–198, 1931.
- [12] J. Hartley Rogers. *Theory of Recursive Functions and Effective Computability*. McGraw-Hill Book Co, 1967.
- [13] M. Hils and F. Loeser. *A first Journey through Logic*, volume 89 of *Student Mathematical Library*. American Mathematical Society, Providence, RI, September 2019.
- [14] W. Hodges. *Model theory*. Cambridge University Press, Cambridge, 1993.
- [15] W. Hodges. *A shorter model theory*. Cambridge University Press, Cambridge, 1997.
- [16] N. Immerman. *Descriptive Complexity*. Graduate Texts in Computer Science, Springer, New York, 1998.
- [17] T. Jech. *Set theory*. Springer Monographs in Mathematics. Springer-Verlag, Berlin, 2003. The third millennium edition, revised and expanded.
- [18] A. Kechris. *Classical descriptive set theory*, volume 156 of *Graduate Texts in Mathematics*. Springer, 1995.
- [19] J. Krajčevič. *Proof complexity*. Cambridge University Press, 2019. Encyclopedia of Mathematics and Its Applications, Vol.170.
- [20] L. Libkin. *Elements of Finite Model Theory*. Springer, 2004.
- [21] Y. V. Matiyasevich. *Hilbert’s Tenth Problem*. MIT Press, Cambridge, Massachusetts, 1993.
- [22] Y. N. Moshovakis. *Notes on Set Theory*. Springer Science and Business Media, 1994.
- [23] C. H. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.
- [24] C. René and D. Lascar. *Logique mathématique : cours et exercices; préface de J.-L. Krivine, 1. Calcul propositionnel, algèbres de Boole, calcul des prédicats*. Axiomes collection de logique mathématique. Masson, Paris, 1993.
- [25] J. B. Rosser. Extensions of some theorems of Gödel and Church. *Journal of Symbolic Logic*, 1:87–91, 1936.
- [26] K. Schütte. *Proof theory*. Springer-Verlag Berlin Heidelberg, 1977.
- [27] S. Shelah and E. Milner. Graphs with no unfriendly partitions. *Special issue ‘a tribute to Paul Erdős’, Hungarian Academy of Sciences*, 1990.
- [28] T. Skolem. Einige Bemerkungen zur axiomatischen Begründung der Mengenlehre. *Matematikkongressen i Helsingfors*, pages 217–232, 1922.

- [29] R. M. Solovay. Provability interpretations of modal logic. *Israel Journal of Mathematics*, 25:287–304, 1976.
- [30] K. Tent and M. Ziegler. *A course in model theory*. Lecture Notes in Logic. Cambridge University Press, 2012.
- [31] A. M. Turing. On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society*, 42:230–265, 1937.
- [32] M. Ziegler. *Mathematische Logik, 2. Auflage*. Birkhäuser, 2017.