Logic II
Lecture Notes

Josef Schicho
RISC Linz
Johannes Kepler University, A-4040 Linz
e-mail: Schicho@risc.uni-linz.ac.at

March 29, 2001

Contents

0 Introduction

There are two motivations for the desire to formalize reasoning. On the one hand, one wants to have some sort of "quality control", in order to check and improve reliability and efficiency of proofs. This was the impetus behind Euclid's efforts to formalize geometry. It was also the motivation of Frege, Russell, Hilbert, Dedekind, ... , to construct a universe of mathematics based on formal systems. Historically, the efforts of axiomatization led to the discovery of antinomies, which in turn increased the necessity to improve on reliability. A more modern example are the efforts of Buchberger to improve the "quality of proofs" of his students in his famous Thinking-Speaking-Writing course.

A completely different goal is the desire to automatically generate or verify theorems. This idea can be traced back to Lullus (around 1300, "ars magna") and Leibniz (1646–1716), who distinguishes decision methods "ars iudicandi" and generation methods "ars inveniendi". Modern computer science adopted this vision as the core of the "Artificial Intelligence" program. Even though limitations of this approach became clear, we have now many useful algorithms. Most notably are Robinson's resolution algorithm and Collins' quantifier elimination algorithm.

The theory of formal calculi of reasoning is called meta-theory, while the theory of relations between formal calculi and their semantics is called model theory. In this course, we discuss selected topics from these two main branches of logics.

Predicate calculus (also called first order logic). We give proofs for Church's result of undecidability of predicate calculus, Gödel's completeness theorem, and Herbrand's theorem.

Formal analysis . We give an axiomatization of the first order theory of the real numbers and the decidability due to Tarski. As an application, we give an axiomatization of geometry (also due to Tarski), which is also decidable. In contrast to this, we discuss constructive analysis in an informal way.

Formal number theory . Two famous negative results - Gödel's incompleteness theorem and the unsolvability of Hilbert's 10th problem - are obtained as immediate consequences of two positive results (which are closely related): any recursively enumerable set is arithmetic (Gödel) and any recursively enumerable set is diophantine (Matijasevic).

For the preparation of these notes, I leaned on the book [16], as far as predicate logic and formal number theory is concerned. The section on formal geometry is based on [10], the section on constructive analysis is based on [3], and the section on Hilbert's 10th problem is based on [15]. Several former and present participants also have contributed to the notes, among them Florina Piroi (completeness theorem), Temur Kutsia (completeness theorem), Cleo Pau (formal number theory), and Igor Rents (Hilbert's 10th problem).

For further study, I recommend [30, 11, 25, 13, 26, 17, 9, 18, 8, 2] for studying predicate logic and [12, 7, 19, 6, 21, 28, 29, 3] for studying the theory of reals and geometry. For automated theorem proving, I refer to the lectures [14] and to [5].

1 Formal Theories

In this section, we introduce the main concepts of logic, draw the distinctions between object theory, metatheory, and semantics, and explain the objectives and the methods of each of these topics. We conclude this section by giving some philosophical positions on the role and the relations between these topics.

A *formal theory* consists of

- linguistical part:
  - symbols (predicate symbols, function symbols, logical connectives, variables, ...
  - terms (described by a syntax)
  - formulas, sentences (described by a syntax)
- logical part:
  - logical axioms
  - inference rules
- material part:
  - material axioms

A *formal theorem* is either an axiom or an immediate consequence of other formal theorems by some inference rule. A **proof** is a sequence of theorems such that each theorem is either an axiom or an immediate consequence of theorems before it.

The purpose of a formal theory is to formally describe reasoning/proving in some informal "real world" theory (number theory, analysis, set theory ...).

Metatheory is reasoning about the formal theory (which is also called *object theory* when the distinction to metatheory is emphasized). Metatheory is again an informal theory.

In contrast to semantics, a metatheory investigates a formal system as formal system, abstracting from the informal theory which is going to be formalized. In finding problems and solutions, it may be guided by the informal system, but in the final statement and in the proof, the use of the interpretation is prohibited.

Typical problems:

- Consistency (does there exists a non-theorem?)
- Completeness (can every formula be either be proven or refuted?)
- Decidability (does there exist an algorithm that decides whether a given formula is a theorem?) [1]

[1]The fact that theorems are constructed by a procedure which is usually recursive—the application of inference rules—makes sure that the set of theorems is recursively enumerable.

- Proof theoretical questions:
  - Do there exist canonical proofs?
  - Given a theorem, how long can the shortest proof be?

Model Theory investigates the relation of the formal system and the informal theory which is going to be formalized. Typical problems:

- Correctness (is every theorem true in the model?)
- Completeness w.r.t. the interpretation, or relative completeness (is every true statement in the model a theorem?)
- Definability (is every statement in the model expressible in the theory?)
- Categoricity (do there exist non-standard models?)

These disciplines are not always strictly separated. For instance, the usual method for showing consistency is to give a model, i.e. a metatheoretic question is given a model theoretic answer.

In the discussion about foundations of mathematics in the beginning of this century, there arose a doubt on the validity of such consistency proofs, because they used the existence of certain mathematical objects. But do these things really exist or is this merely an illusion? It is interesting to listen to the answers given by various schools.

The viewpoint of *formalism* (Hilbert) is the following. In mathematics, an object exists - by definition - if it can be described consistently. As far as "mathematical truth" is concerned, we need nothing more than a formal theory. Truth is then defined as provability in this theory.

From the formalistic point of view, consistency proofs are a fundamental task, in order to guarantee existence of mathematical objects. But of course, it is not allowed to prove consistency by model theory, because we would fall into a vicious circle. Thus, we ask for a consistency proof with purely meta-theoretic methods, accepting only a small subset of mathematics as the tools for such a proof. Hilbert and his followers required more specifically, that the meta-theory should be restricted to "finitary means".

In the original sense, Hilbert's program broke down: there cannot be a finitary proof of the theory of natural numbers, as a consequence of Gödel's second incompleteness theorem.

On the other hand, there is the viewpoint of *intuitionism* (Brouwer): certain mathematical object do exist because they exist in the human imagination. In order to justify "mathematical existence", one has to establish the connection to this intuition. Mathematical proofs are as true as the conviction they carry.

A formalization of any mathematical theory is just an abstract description of the reasoning/proving within that theory. The formalization is correct only if it follows intuition. Consistency alone is not sufficient: 'An incorrect theory which is not stopped by a contradiction is none the less incorrect, just as a criminal policy unchecked by a reprimanding court is none the less criminal.' In this sense, Brouwer critizized many mathematical theories as not related to any human imagination; these were born just by overtaking "logical principles" from another theory without justification.

In particular, Brouwer attacks the universal use of the principle of excluded middle, because it is not convincing when applied to undecided statements. It should be pointed out that Brouwer did not claim that the principle of excluded middle is false, he just stated that it is not true universally.

The viewpoint of *platonism* claims that a formalization of a mathematical theory is an approximate description of a mathematical world which is independant of our reasoning. In this world, every statement is either true or false, no matter of our knowledge of it. The platonist therefore upholds the principle of excluded middle. He/she is able to prove more theorems than the intuitionist, but at the same time he is forced to admit that there are theorems which can never be proven. They are just true (in the mathematical world), but we finite brains can never have the insight of recognizing their truth. And this is provably so.

The most popular viewpoint among mathematicians is the *pragmatic*. People believe that the abandonment of the principle of excluded middle would cripple mathematics, as many theorems cannot be proven and for others, the proofs become more difficult. They are not willing to pay such a prize just because of metaphysical considerations. Therefore the principle of excluded middle is accepted.

There is, however, a modern successor of intuitionism without any metaphysical ambitions. The *constructivists* point out that any theorem proven without the excluded middle has an algorithmic meaning, and the proof automatically entails an algorithm, at least implicitly. Thus, a proof with intuitionistic logic is preferred to a proof with classical logic.

2 Theory of Recursion

In this section, we give a survey on the theory of recursion, including the existence of recursively enumerable sets which are not recursive.

Intuitively, a recursive function is a function from $\mathbb{N}^k$ to $\mathbb{N}$ which can be computed by a finite computer program executed on a virtual machine with infinite memory. To make this definition more formal, one needs a theoretical model for computation.

Since this model has only a theoretical purpose, one wants to keep the set of primitive operations as small as possible. A well-known model is the one of Turing machines, which can be imagined as a single read-write button attached to an infinite memory strip.

Another model is the model of recursion. Here one tries to define computable functions in a machine-independant way, reflecting the idea that one should not write programs for existing machines but devise machines so that they are able to execute existing programs (Dijkstra).

We start by defining base functions. These are:

1. the successor function $' : \mathbb{N} \to \mathbb{N},\ x' := x + 1$;
2. the constant functions $c_q^k : \mathbb{N}^k \to \mathbb{N},\ c_q(x_1, \ldots, x_k) := q$;
3. the projection functions $p_i^k : \mathbb{N}^k \to \mathbb{N},\ p_i(x_1, \ldots, x_k) := x_i$.

Now, let $g : \mathbb{N}^{k-1} \to \mathbb{N},\ h : \mathbb{N}^{k+1} \to \mathbb{N}$ be two functions. Then the primitive recursion f of g and h is defined as the unique function satisfying

- $f(0, x_2, \ldots, x_k) = g(x_2, \ldots, x_k)$,
- $f(x', x_2, \ldots, x_k) = h(x, f(x, x_2, \ldots, x_k), x_2, \ldots, x_k)$.

A function is called primitive recursive iff it can be obtained from base functions by compositions and primitive recursions. A construction of f by compositions, primitive recursions, and base functions, is called a program for f.

Example 1 Addition is primitive recursive, because we have the primitive recursion

$$0 + y = y, x' + y = (x + y)',$$

which gives $+$ as the primitive recursion of p_1^1 and the composed function $(p_2^3)'$. In other words, $\mathrm{PrimRec}[\mathrm{p}_1^1, (\mathrm{p}_2^3)']$ is a program for $+$.

Multiplication is primitive recursive because

$$0 \cdot y = 0, x' \cdot y = (x \cdot y) + y.$$

This representations has the program $\mathrm{PrimRec}[\mathrm{c}_0^1, \mathrm{PrimRec}[\mathrm{p}_1^1, (\mathrm{p}_2^3)'](\mathrm{p}_2^3, \mathrm{p}_3^3)]$.

Exercise 1 Show that the functions exponentiation, predecessor, difference (defined as zero if the second argument is bigger than the first), minimum, maximum, zerotest, symmetric difference, remainder, quotient are all primitive recursive.

A predicate $P(x_1, \dots, x_n)$ is called primitive recursive iff the characterstic function

$$\chi_P(x_1, \dots, x_n) = \begin{cases} 1 & \text{if } P(x_1, \dots, x_n) \text{ is true,} \\ 0 & \text{otherwise} \end{cases}$$

is primitive recursive. A subset $L \subset \mathbb{N}$ is primitive recursiv iff the predicate "is contained in L" is primitive recursive.

For examples, the predicates "less than" (binary), "divisible" (binary), "prime" (unary) are all primitive recursive.

Let $g : \mathbb{N}^{k+1} \to \mathbb{N}$ be a function. Then the minimalization f of g is defined by

$$f(x_1, \dots, x_n) = \begin{cases} \text{the smallest } x \text{ such that } g(x, x_1, \dots, x_n) = 0\text{, if exists;} \\ \text{undefined otherwise.} \end{cases}$$

The minimalization is a partial function from $\mathbb{N}^k$ to $\mathbb{N}$.

If g is itself a partial function, then $f(x_1, \dots, x_n)$ is defined only in the case when there is an x such that $g(x, x_1, \dots, x_n) = 0$ and $g(y, x_1, \dots, x_n)$ is defined for all $y < x$. Composition and primitive recursion is generalized to partial functions in the obvious way.

The partial function is called partial recursive iff it can be obtained from the base functions by compositions, primitive recursions, and minimalizations. A partial function which is defined everywhere is called a recursive function.

There are recursive functions which are not primitive recursive, for example the Ackermann function, which is defined by the following double recursion:

- $ack(0, y) = y''$,
- $ack(x', 0) = ack(x, 1)'$,
- $ack(x', y') = ack(x, ack(x', y))$.

It is obvious that for each fixed x, the function $y \mapsto ack(x, y)$ is primitive recursive.

Theorem 1 *The binary Ackermann function is not primitive recursive.*

Proof: The proof proceeds in a series of claims, each depending on the previous claims.

Claim 1: $ack(x, y) > y$. This is proved by double induction. For $x = 0$, the claim is true by definition. Assume the claim holds for $x = x_0$. We prove that it holds for $x = x_0'$, by induction on y. For $y = 0$, the claim is true by definition. Assume that the claim holds for $y = y_0$. Then

$$ack(x_0', y_0') = ack(x_0, ack(x_0', y_0)) \geq ack(x_0', y_0) + 1 \geq y_0 + 2 > y_0',$$

where we have used both induction hypotheses in the inequality.

Claim 2: $ack(x, y') > ack(x, y)$. This is proved by case distinction. If $x = 0$, then statement is equivalent to $y''' > y''$, which is true. If $x = z'$, then we get

$$ack(z', y') = ack(z, ack(z', y)) > ack(z', y)$$

by claim 1.

Claim 3: ack is strictly monotonic in the second argument. This is immediate from claim 2.

Claim 4: $ack(1, y) = 2y + 4$. Proof by induction on y.

Claim 5: $ack(x, y') < ack(x', y)$. Proof by double induction. For $x = 0$, the claim is equivalent to $y + 3 < 2y + 4$ by claim 4, and this is clearly true.

Assume the claim holds for $x = x_0$. We prove that it holds for $x = x_0'$, by induction on y. For $y = 0$, the claim is true by definition. Assume that the claim is true for $y = y_0$. Then

$$ack(x_0', y_0'') = ack(x_0, ack(x_0', y_0')) < ack(x_0', ack(x_0', y_0') - 1) <$$

$$ack(x_0', ack(x_0'', y_0)) = ack(x_0'', y_0').$$

Here we used the induction hypothesis on x and, in the second inequation, claim 3 and the inequality

$$ack(x_0', y_0') - 1 < ack(x_0', y_0') < ack(x_0'', y_0)$$

that follows from the induction hypothesis on y.

Claim 6: $ack(x, y) < ack(x', y)$. Immediate from claim 2 and claim 5.

Claim 7: ack is strictly monotonic in the first argument. This is immediate from claim 6.

Claim 8: For any x_1, x_2, we have $ack(x_1, ack(x_2, y)) < ack(x_3, y)$, where $x_3 = \max(x_1 + 2, x_2 + 1)$. This is so because

$$ack(x_1, ack(x_2, y)) \leq ack(x_3 - 2, ack(x_3 - 1, y)) = ack(x_3 - 1, y') < ack(x_3, y),$$

where the two inequalities follow from claim 7 and claim 5.

Claim 9: For any n-ary primitive recursive function f, there is a number c such that

$$f(x_1, \ldots, x_n) < ack(c, x_1 + \cdots + x_n)).$$

(We also say that f is bounded by c.) For the constant functions, the successor function, and the projection functions, the claim is true: choose $c := 0$ for the successor function and for the projections, and choose c to be the corresponding constant in the case of constant functions.

Let g be an n-ary function bounded by c, and let $h_1, \ldots, h_n$ be m-ary functions bounded by d. Then we can bound the composition by

$$f(h_1(x_1, \ldots, x_m), \ldots, h_n(x_1, \ldots, x_m)) <$$

$$ack(c, h_1(x_1, \ldots, x_m) + \cdots + h_n(x_1, \ldots, x_m)) <$$

$$ack(c, n\ ack(d, x_1 + \cdots + x_m)) < ack(e, x_1 + \cdots + x_m)$$

by claim 3 and claim 8.

Let g be $n-1$-ary bounded by c and h be $n+1$-ary bounded by d. Then we can bound the primitive recursion f by $e := \max(c, d+2)$. We prove this by induction on the main argument x of f. If $x = 0$, then

$$f(0, x_2, \ldots, x_n) = g(x_2, \ldots, x_n) < ack(c, \max(x_2, \ldots, x_n)),$$

and the bound follows from the assumption. Suppose the bound holds for $x = x_0$. Then we have

$$f(x_0', x_2, \ldots, x_n) = h(x_0, f(x_0, x_2, \ldots, x_n), x_2, \ldots, x_n) <$$

$$ack(d, \max(x_0, f(x_0, x_2, \ldots, x_n), x_2, \ldots, x_n)) <$$

$$ack(e-2, ack(e-1, \max(x_0, x_2, \ldots, x_n)) = ack(e-1, \max(x_0, x_2, \ldots, x_n)') <$$

$$ack(e, \max(x_0', x_2, \ldots, x_n))$$

by claim 1, claim 3, and claim 8.

If the Ackermann function were primitive recursive, then we could bound it by some number, say c. Then we get

$$ack(c, c) < ack(c, \max(c, c)) = ack(c, c),$$

a contradiction.

The following theorem is interesting, but it will not be used later.

Theorem 2 *Any partial recursive function f has a program with only a single minimalization.*

Proof: For any function $h(x_1, \ldots, x_n)$, define the bounded minimalization m of h by

$$m(x_1, \ldots, x_n) = \begin{cases} \text{the smallest } y < x_1 \text{ such that } h(y, x_2, \ldots, x_n) = 0, \text{ if exists;} \\ x_1 \text{ otherwise.} \end{cases}$$

One can show that the bounded minimalization of h can be obtained by primitive recursion and composition from h and the base functions. Therefore, the bounded minimalization of a primitive recursive function is primitive recursive.

Now, let f be any partial recursive function. Let f_0 be the primitive recursion with one more argument, obtained by replacing all minimalizations by a bounded minimalization. Let f_1 be the function obtained from f_0 that behaves like f_0 but "propagates the error due to bounding", i.e. it outputs x_1 whenever x_1 was obtained as the answer of a bounded minimalization. It is easy to show that also f_1 is primitive recursive. Let g_1 be the minimalization of f_1. Then it holds that $f(x_1, \ldots, x_n) = f_1(g(x_1, \ldots, x_n), x_1, \ldots, x_n)$.

At this point, it should be intuitively clear that "recursion" is a universal programming language like Turing machines, ANSI-C, or Fortran 77. Since the number of available primitives is small, real programming would be cumbersome. But in principle we get the same computational power.

A predicate or subset of natural numbers is called recursively enumerable iff it is the domain of definition (abbreviation: DEF) of a partial recursive function. A predicate or subset P is called recursive iff both P and the complement of P are recursively enumerable.

Theorem 3 **a)** *A predicate is recursive iff its characteristic function is recursive.*

b) *A function is recursive iff its graph (a predicate with arity one higher) is recursive.*

c) *A partial function is recursively enumerable iff its graph is recursively enumerable.*

Proof: (a), $\Rightarrow$: let X_y and X_n be two programs with domains of definition P and the complement of P. The following program with input $x_1, \ldots, x_n$,

> run X_y and X_n in parallel; if X_y stops first, then return y; if X_n stops first, then return n

returns a result for all input values, and computes the characteristic function.

(a), $\Leftarrow$: let X be a program for the characteristic function of the predicate P (in other words, a program deciding P). the following program with input $x_1, \ldots, x_n$,

run X on $x_1, \ldots, x_n$;
if the result is 1, then return 1;
else loop forever

has the elements satisfying P as its domain of definition. Similarily, one constructs a program with the complement as its domain of definition.

(b), $\Rightarrow$: let X be a program computing f. The following program with input $x_1, \ldots, x_n, y$,

compute $f(x_1, \ldots, x_n)$ using X;
if the result is y, return 1;
else return 0

computes the graph of f.

(b), $\Leftarrow$: let X be a program computing the graph of f. The following program with input $x_1, \ldots, x_n$,

for i from 0 to infinity
if X applied to $x_1, \ldots, x_n, i$ gives 1, return i

always terminates and computes f.

(c), $\Rightarrow$: let X be a program computing f. The following program with input $x_1, \ldots, x_n, y$,

compute $f(x_1, \ldots, x_n)$ using X;
if the result is y, return 1;
else loop forever

has the graph of f as its domain of definition.

(c), $\Leftarrow$: let X be a program with the graph of f as its domain of definition. Let X' be the program with one more argument obtained by replacing all minimalizations by bounded minimalizations. The following program with input $x_1, \ldots, x_n$

for i from 0 to infinity
for j from 0 to i
apply X' to $i - j, x_1, \ldots, x_n, j$
if the result is 1, return j

computes f.

Interpreters are programs that have programs as an input and simulate the behavior of the input program. In order to express an interpreter as a recursive function, we need a coding function $\mathcal{C}$ from the set of all programs to $\mathbb{N}$. The precise definition of the function is not really relevant (except for the interpreter), provided that it fulfills some requirements:

- the coding is unique, i.e. $\mathcal{C}$ is injective;

- given a number, we can get its outermost constructor. In other words, the predicates "is a code of a composite" and so on are all recursive;
- we can parse the code. More precisely: given the code of a program, we can compute the codes of all its subprograms, i.e. the corresponding functions are recursive;
- the codes of the subprograms are smaller than the codes of the main program.

For all reasonable codes one can think of (e.g. ASCII code), the requirements are fulfilled even with primitive recursive destructors.

Theorem 4 *The binary predicate "m is the code of program of an n-ary partial recursive function" is recursive.*

Proof: Write a parser (a program that parses codes for programs). Remark: if the coding has primitive recursive destructors, then one can write a primitive recursive parser, so that the binary predicate above becomes even primitive recursive.

Theorem 5 *For any n, the $n+1$-ary partial function*

$$i_n(z, x_1, \ldots, x_n) = \begin{cases} f(x_1, \ldots, x_n) & \textit{if } z \textit{ is the code of program} \\ & \textit{for a partial recursive function } f \\ \textit{undefined otherwise.} & \end{cases}$$

is partial recursive.

Proof: Write an interpreter.

Theorem 6 *There exists a recursively enumerable set which is not recursive.*

Proof: Let $R \subset \mathbb{N}$ be the domain of definition of the function $x \mapsto i_1(x, x)$. Then R is recursively enumerable. We show that the complement is not recursive.

Suppose indirectly, that there exists a unary partial recursive function f with $DEF(f) = \mathbb{N} - R$. By definition, $x \in DEF(f)$ iff either x is the code of a program X such that $x \notin DEF(X)$, or x is not the code of a program of a unary partial function.

Let y be the code of a program Y for f. Then $y \in DEF(Y)$ iff $y \notin DEF(Y)$, and this is a contradiction.

3 Equational Theories

An equational theory is has the following elements.

Linguistical part:

- variables (infinitely many)
- function symbols (including constant symbols)
- one binary predicate "="

Terms are words in the symbols above according to the following syntax:

- a variable is a term;
- if $t_1, \ldots, t_n$ are terms, and f is a n-ary function symbol, then $f(t_1, \ldots, t_n)$ is a term.

If t_1, t_2 are terms and x is a variable, then we write $t_1[x := t_2]$ for the term obtained by replacing all occurences of x by t_2. Sometimes, we write more informally $t_1(x)$ for a term in which we want to substitute x, and $t_1(t_2)$ for the result of the substitution.

Logical part:

- Axioms of the form $t_1 = t_2$;
- Inference rules:
  1. $t = t$;
  2. $t_1 = t_2 \vdash t_2 = t_1$;
  3. $t_1 = t_2, t_2 = t_3 \vdash t_1 = t_3$;
  4. $s_1 = t_1, \ldots, s_n = t_n \vdash f(s_1, \ldots, s_n) = f(t_1, \ldots, t_n)$;
  5. $t_1 = t_2 \vdash t_1[x := t] = t_2[x := t]$.

All theorems are equations of the type $t_1 = t_2$. If $t_1 = t_2$ is provable in the theory, then we also say that t_1 and t_2 are convertible. Clearly, convertibility is an equivalence relation. A theory is consistent if there are at least two terms which are not convertible.

Example 2 The equational theory of groups has the binary function symbol mult, unary function symbol inv, and constant symbol one. The axioms are the following.

1. $\mathrm{mult}(\mathrm{mult}(x, y), z) = \mathrm{mult}(x, \mathrm{mult}(y, z))$;
2. $\mathrm{mult}(x, \mathrm{one}) = x$;
3. $\mathrm{mult}(\mathrm{one}, x) = x$;

4. $\mathrm{mult}(\mathrm{x}, \mathrm{inv}(\mathrm{x})) = \mathrm{one}$;

5. $\mathrm{mult}(\mathrm{inv}(\mathrm{x}), \mathrm{x}) = \mathrm{one}$.

Let $\mathcal{T}$ be an equational theory. A structure for $\mathcal{T}$ is a set S, together with a function $\mathbf{f} : S^n \to S$ for each n-ary function symbol f. An assignment is a function from the set of variables to S. An interpretation is a structure together with an assignment.

For any term, one can define its value under some interpretation $\mathcal{I}$ recursively. An equation is true in some interpretation iff the left hand side and the right hand side evaluate to the same element in S. A structure is a model for $\mathcal{T}$ iff all axioms are true for all assignments. For instance, the models for the example above are precisely the groups.

It is clear that equational theories are semantically consistent: if an equation is provable, then it is true in every model under any interpretation. The converse (semantical completeness) is also true.

Theorem 7 *If an equation is true for all models and all assignments, then it is provable.*

Proof: Here is a construction of a model M_0 and an assignment A_0, for which only the provable equations are true.

Let H be the Herbrand structure of $\mathcal{T}$, defined as the set of all terms in $\mathcal{T}$. The functions $\mathbf{f} : H^n \to H$ are defined in the obvious way: $\mathbf{f}(t_1, \ldots, t_n) := f(t_1, \ldots, t_n)$.

Denote the convertibility relation by $\sim$. Then all functions $\mathbf{f}$ are well-defined for equivalence classes. Thus, $H/\sim$ is a structure for $\mathcal{T}$.

Let A be an assignment and let $t_1 = t_2$ be an axiom. The assignment defines a substitution S, and by inference rule 4 we have that $t_1[S] = t_2[S]$. Hence the values of t_1 and t_2 are equal in the structure $H/\sim$ and assignment A. Therefore $M_0 := H/\sim$ is a model for $\mathcal{T}$.

The assignment A_0 maps each variable to itself.

Now, let $t_1 = t_2$ a formula which is true in this model and assignment. Then t_1 and t_2 are convertible, i.e. $t_1 = t_2$ is provable in $\mathcal{T}$.

The class of all models for a fixed theory is called a variety, and it has a lot of interesting properties (closed under products, subobjects, quotients). Such structures are studied in *universal algebra*.

In order to analyze convertibility, it is often useful to consider an asymmetric variant of it—reduction.

Let T be the set of terms of a theory $\mathcal{T}$. A reduction on T is just a binary relation on T, written $t_1 \to t_2$.

A reduction is called compatible (with the function symbols) iff $t_1 \to t_2$ implies $f(s_1, \ldots, s_{i-1}, t_1, s_{i+1}, \ldots, s_n) \to f(s_1, \ldots, s_{i-1}, t_2, s_{i+1}, \ldots, s_n)$. A reduction is called substitutive if $t_1 \to t_2$ implies $t_1[x := t] \to t_2[x := t]$.

The reflexive/transitive/symmetric/compatible/substitutive closure of $\rightarrow$ is the smallest relation extending $\rightarrow$. The reflexive transitive closure is also denoted by $\stackrel{*}{\rightarrow}$, and the reflexive transitive symmetric closure is denoted by $\stackrel{*}{\leftrightarrow}$.

Given a theory, one gets a reduction by replacing each axiom $t_1 = t_2$ by a relation $t_1 \rightarrow t_2$, which is also called a rewrite rule. The reduction $\rightarrow$ is defined as the compatible substitutive closure of the set of rewrite rules. Clearly, compatibility is the reflexive transitive symmetric closure of $\rightarrow$.

A reduction $\rightarrow$ has the diamond property iff for all t, t_1, t_2 such that $t \rightarrow t_1$ and $t \rightarrow t_2$, there exists t_3 such that $t_1 \rightarrow t_3$ and $t_2 \rightarrow t_3$.

A reduction $\rightarrow$ is Church-Rosser (or confluent) iff $\stackrel{*}{\rightarrow}$ has the diamond property.

A term t is called a normal form iff there is no t' such that $t \rightarrow t'$. A term t_1 is called a normal form of t_2 iff it is a normal for and $t_1 \stackrel{*}{\leftrightarrow} t_2$.

Lemma 1 *If $\rightarrow$ has the diamond propery, then $\stackrel{*}{\rightarrow}$ has the diamond property.*

Proof: By a diagram chase (see [1], p 59).

Theorem 8 *Let $\rightarrow$ be a Church-Rosser reduction. Let $\stackrel{*}{\leftrightarrow}$ be its symmetric transitiv closure. Let $t_1 \stackrel{*}{\leftrightarrow} t_2$. Then there exists t_3 such that $t_1 \stackrel{*}{\rightarrow} t_3$ and $t_2 \stackrel{*}{\rightarrow} t_3$.*

Proof: By another diagram chase (see [1], p 54).

Corollary 1 *Let $\rightarrow$ be Church-Rosser. Then the following hold.*

a) *If t_1 has a normal form t_2, then $t_1 \stackrel{*}{\rightarrow} t_2$.*

b) *A term can have at most one normal form.*

Proof: (a): by theorem 8, t_1 and t_2 both reduce to common reductum t_3. Because t_2 is a normal form, it follows $t_2 = t_3$.

(b): suppose that there are two equivalent normal forms t_1, t_2. By (a), we have a reduction $t_1 \stackrel{*}{\rightarrow} t_2$. Since t_1 is a normal form, it follows $t_1 = t_2$.

Suppose that $\rightarrow$ is a reduction such that $\stackrel{*}{\leftrightarrow}$ is convertibility, for instance the compatible substitutive closure of the rewrite rules. Suppose moreover that $\rightarrow$ is Church-Rosser. Given two terms t_1, t_2, one can try to reduce each term until no more reduction is possible, in order to get normal forms for t_1 and t_2. Then compatibility may be decided by comparing the normal forms on a syntactical level.

In the case of λ-calculus, the above strategy goes down and gives an algorithm for recognizing compatibility. The algorithm may not terminate, in the case where the terms do not have a normal form.

Other cases are particularily nice: when reduction is a finite process.

A reduction path is a sequence

$$t_0 \to t_1 \to t_2 \ldots.$$

A reduction is called noetherian (or strongly normalizing) iff every reduction path is finite.

In a noetherian reduction, any term has a normal form. Thus, we have a decision algorithm for compatibility when a reduction is noetherian and Church-Rosser (under the assumption that one can decide whether one-step reduction is possible).

Moreover, the noetherian property allows to weaken the second assumption:

A reduction $\to$ is called weakly Church-Rosser (or local confluent) iff for all t, t_1, t_2 such that $t \to t_1$ and $t \to t_2$, there exists t_3 such that $t_1 \xrightarrow{*} t_3$ and $t_2 \xrightarrow{*} t_3$.

Theorem 9 (Newman) *If a reduction is noetherian and weakly Church-Rosser, then it is Church-Rosser.*

Proof: Call an element ambiguous iff it reduces to two different normal forms. We give an indirect proof for the non-existence of ambiguous elements.

Let t_1 be ambiguous with the two distinct normal forms t_2 and t_3. Obviously, t_1 is not a normal form, therefore the reductions to t_2 and t_3 are at least of length one. Therefore we have elements t_4 and t_5, obtained from t_1 by a one step reduction, t_4 in the path to t_2 and t_5 on the path to t_3. By the weak Church-Rosser property, there is an element t_6 such that t_4 and t_5 both reduce to t_6. Let t_8 be a reduction of t_7 to a normal form.

Clearly, we have either $t_2 \neq t_8$ or $t_3 \neq t_8$. If $t_2 \neq t_8$, then t_4 is ambiguous, since it reduces to the normal forms t_2 and t_8. If $t_3 \neq t_8$, then t_5 is ambiguous, since it reduces to the normal forms t_3 and t_8. In any case, one finds an ambiguous reductum.

Since t_1 was an arbitrary ambiguous element, we can repeat the construction and obtain a reduction path of infinite length, contradicting noetherianity.

Suppose that we are in the noetherian case, but our reduction is not weakly Church-Rosser: there are t, t_1, t_2 such that $t \to t_1$ and $t \to t_2$, and t_1 and t_2 have no common reductum. Then one can try to force the weak Church-Rosser property by adjoining $t_1 \to t_2$ as a new rewrite rule.

A priori, the search space for counter-examples to weak Church-Rosser is infinite, even if there only finitely many rewrite rules. The theorem of Knuth-Bendix reduces the search space considerably.

Two terms t_1, t_2 are called unifiable if there exists a substitution θ (called unifier) with the property $t_1[\theta] = t_2[\theta]$. A *most general unifier* is a unifier σ such that for any unifier θ there exists a substitution λ with $\theta = \sigma\lambda$. Most general unifiers are unique up to a renaming of variables.

Theorem 10 (Unification) *If t_1 and t_2 are unifiable, then there exists a most general unifier.*

Proof: By induction over the number of variables in t_1 and t_2.

If there are no variable, the theorem is trivially true.

The *disagreement set* is the set of the two subexpressions beginning with the first symbol where t_1 and t_2 disagree. Any unifier must unify the disagreement set; this is possible only if one of the expressions is a variable x and the other is a term t which does not contain x. Let $\alpha := \{x \leftarrow t\}$, $t_1' := t_1\alpha$, $t_2' := t_2\alpha$. Then any unifier θ can be written as $\theta = \alpha\theta'$, where θ' is a unifier of t_1' and t_2'. If t_1' and t_2' are not unifiable, then t_1 and t_2 are not unifiable and the theorem is true. Otherwise, there exists a most general unifier σ' for t_1' and t_2' by the induction hypothesis, because t_1' and t_2' do not contain the variable x. Then, $\sigma := \alpha\sigma'$ is a most general unifier for t_1, t_2.

Exercise 2 Determine whether $g(x, y, z)$ and $g(f(y, y), f(z, z), f(x, x))$ are unifiable. If yes, obtain a most general unifier.

Let $s_1 \to t_1$ and $s_2 \to t_2$ be two rewrite rules. Let p be a subterm of s_2 which is not a variable, and suppose that s_1 and p are unifiable. Then the two rewrite rule and the subterm occurence define a critical pair, in the following way:

First, we rename variables so that the two rewrite rules habe no variable in common.

Let σ be the most general unifier of p and s_1. Then $p[\sigma]$ is a subterm of $s_2[\sigma]$. Let $p_1 := s_2[\sigma]\{p[\sigma]/t_1[\sigma]\}$ be the result of replacing this subterm by $t_1[\sigma]\}$. (Note that p_1 depends not just on the subterm, but also on the occurence of the subterm in the whole term!) Then the critical pair is the pair of terms (p_1, p_2), where $p_2 := t_2[\sigma]$.

Since $\to$ (the reduction defined by the rewrite rules) is compatible and substitutive, we have that $s_2 \to p_1$ and $s_2 \to p_2$, and consequently $p_1 \overset{*}{\leftrightarrow} p_2$.

Theorem 11 (Knuth-Bendix) *A system of rewrite rules defines a weakly Church-Rosser reduction iff for any critical pair (p_1, p_2) there exists a term q such that $p_1 \overset{*}{\to} q$ and $p_2 \overset{*}{\to} q$.*

Proof: See [33], p 53f.

This is the idea of the Knuth-Bendix completion procedure for establishing a system of rewrite rules with unique normal form.

1. form all critical pairs;
2. for each critical pair (p_1, p_2), reduce p_i to a normal form q_i;

3. if $q_1 \neq q_2$, add $q_1 \to q_2$ or $q_2 \to q_1$ as a new rewrite rule;

4. repeat;

Noetherianity is a crucial point in this procedure. One has to make sure somehow that noetherianity is also preserved when new rules are added.

Example 3 We start with the rewrite rules obtained from the axioms in example 2.

In order to ensure noetherianity, we observe that the left hand side is always shorter than or equal to the right side. In case of equal length, the reductions always shift function symbols to the right. Whenever we add new rules, we orient the critical pair so that these are preserved.

The left side of the first rewrite rules unifies with its subterm $\mathrm{mult}(\mathrm{x},\mathrm{y})$. After renaming variables, we look for the most general unifier of $\mathrm{mult}(\mathrm{mult}(\mathrm{x},\mathrm{y}),\mathrm{z})$ and $\mathrm{mult}(\mathrm{u},\mathrm{v})$. It is $(u := \mathrm{mult}(\mathrm{x},\mathrm{y}), \mathrm{v} := \mathrm{z})$. Then the critical pair is

$$\mathrm{mult}(\mathrm{mult}(\mathrm{x},\mathrm{mult}(\mathrm{y},\mathrm{z})),\mathrm{w}), \mathrm{mult}(\mathrm{mult}(\mathrm{x},\mathrm{y}),\mathrm{mult}(\mathrm{z},\mathrm{w}))).$$

By applying the first rewrite rule twice to the first and once to the second element, one obtains the normal form

$$\mathrm{mult}(\mathrm{x},\mathrm{mult}(\mathrm{y},\mathrm{mult}(\mathrm{z},\mathrm{w}))).$$

Thus, we have no new rewrite rule for this critical pair.

The left sides of the third and fourth rewrite rule are unifiable. The most general unifier of $\mathrm{mult}(\mathrm{one},\mathrm{x})$ and $\mathrm{mult}(\mathrm{y},\mathrm{inv}(\mathrm{y}))$ is $(x := \mathrm{inv}((\mathrm{one})), \mathrm{y} := \mathrm{one})$. We obtain the critical pair $(\mathrm{inv}((\mathrm{one})), \mathrm{one})$, where both sides are already normal forms. Thus, we have to add the new rewrite rule

$$\mathrm{inv}((\mathrm{one})) \to \mathrm{one}.$$

Continuing this way, one obtains a system of 12 rewrite rules which defines a reduction with unique normal form.

Of course, it is not guaranteed that Knuth-Bendix completion terminates.

For a more complete treatment of the Knuth-Bendix and related methods, we refer to [31]. A general framework of "critical pair completion" strategies, which also includes Buchberger's Gröbner bases algorithm [4], can be found in [27].

There are undecidable equational theories (e.g. λ-calculus). The intrinsical difficulty is not substitution (because here we have unification) but compatibility. A well-known problem which is closely related to equational theories is the word problem.

Let Σ be an alphabet of constants. We consider the terms formed by Σ and an associative binary function which we write as concatenation (omitting brackets). Let Γ be a set of equations without variables. Then the word problem for Γ is the following: given an equation $s = t$ without variables, decide if $s = t$ is provable in the theory with axioms Γ (and associativity, of course).

Theorem 12 (Post,Markov) *There is a finite set of equations with an undecidable word problem.*

Proof: See [22].

4 Predicate Logic

First order predicate logic is probably the system which is most studied among all logical calculi. In this section, we will introduce the formal theory and prove some first meta-theorems about it.

Linguistical part:

- predicate symbols, function symbols
- connectives: $\rightarrow$, $\neg$, $\leftrightarrow$, $\wedge$, $\vee$, $\top$, $\bot$
- variables
- quantifiers: $\forall$, $\exists$
- terms, formulas

The binding conventions for conncetives and quantfiers are as follows:

1. quantifiers bind stronger than connectives
2. $\neg$ binds stronger than binary connectives
3. $\wedge$ and $\vee$ bind stronger than $\rightarrow$
4. $\rightarrow$ binds stronger than $\leftrightarrow$
5. $\rightarrow$ binds from right to left, i.e. $A \rightarrow B \rightarrow C$ parses as $A \rightarrow (B \rightarrow C)$

The set of predicate symbols and function symbols together with the arity information is called the signature of the predicate calculus.

An occurence of a variable x in a formula A is said to be *bound* iff the occurence is in a quantifier $\forall x$ or $\exists x$ or in the scope of a quantifier $\forall x$ or $\exists x$. Otherwise, the occurence is called *free*.

A term t is said to be *free for* x *in* A iff no free occurence of x in A is in the scope of a quantifier where the quantified variable occurs in t.

In order to substitute a term t for a variable x in A, we require that t be free for x in A. Otherwise, some part of the term would become bound, and this may change meaning; for example, in the formula $\exists x x = y$, the term $x + 1$ is not free for y. Indeed, we see that the formula is true for all y (with the usual meaning of "="), but substitution would yield $\exists x x = x + 1$, which is false.

A formula is called *closed* (or: a sentence) iff it has no free variables.

Logical part:

- Axioms:
  1. $A \rightarrow B \rightarrow A$
  2. $(A \rightarrow B \rightarrow C) \rightarrow (A \rightarrow B) \rightarrow (A \rightarrow C)$

3. $(\neg A \to \neg B) \to (B \to A)$
4. $\forall x A(x) \to A(t)$, if x is free for t in $A(x)$
5. $\forall x(C \to A(x)) \to C \to \forall x A(x)$, if x is not free in C
6. $(A \to B) \to (B \to A) \to (A \leftrightarrow B)$
7. $(A \leftrightarrow B) \to (A \to B)$
8. $(A \leftrightarrow B) \to (B \to A)$
9. $(A \vee B) \leftrightarrow (\neg A \to B)$
10. $(A \wedge B) \leftrightarrow \neg(A \to \neg B)$
11. $\bot \leftrightarrow (A \wedge \neg A)$
12. $\top \leftrightarrow (A \to A)$
13. $\exists x A(x) \leftrightarrow \neg \forall x \neg A(x)$

- Inference rules:
- $A \to B, A \vdash B$ (modus ponens)
- $A(x) \vdash_x \forall x A(x)$ (generalization)

We say that the variable x is varied in the inference rules 2.

The letters A, B, C above stand for arbitrary formulae, so the axioms are actually axiom schemes. The actual axioms are the instantions of these schemes, which are infinitely many. Note that it is decidable whether a formula is an axiom, and whether a formula can be infererred from one or two other by modus ponens or by generalization.

The consequence relation $\vdash$:

$$A_1, \ldots, A_r \vdash C$$

iff there exists a sequence $B_1, \ldots, B_n = C$ (called derivation), such that each B_i is either an axiom, or one of the A_i, or an immediate consequence of some of $B_1, \ldots, B_{i-1}$ by the inference rules. If a variable is not varied in the derivation, then we say that it is held constant.

Here is a proof of $A \to A$.

1. $A \to (A \to A) \to A$ — axiom 1
2. $(A \to (A \to A) \to A) \to (A \to A \to A) \to (A \to A)$ — axiom 2
3. $(A \to A \to A) \to (A \to A)$ — modus ponens from 2 and 1
4. $A \to A \to A$ — axiom 1
5. $A \to A$ — modus ponens from 3 and 4

We start with a useful meta-theorem.

Theorem 13 (Deduction theorem) *If $\Gamma, A \vdash B$ with all free variables in A held constant, then $\Gamma \vdash A \to B$.*

Proof: We give a proof by induction on the length k of the derivation $\Gamma, A \vdash B$.

If $k = 1$, then we distinguish two cases.

Case 1: B is an axiom or a formula in Γ. Then one can derive $A \to B$ by B and the axiom $B \to A \to B$.

Case 2: B is A. See above exercise.

For the inductive step, suppose that there exists a derivation of B of length $k + 1$.

If B is an axiom, a formula in Γ, or A, then the theorem follows as in the base case $k = 1$.

Suppose that B is obtained by modus ponens as an immediate consequence of two formulas C and $C \to B$, which occur earlier in the proof. By induction hypothesis, the formulas $A \to C$ and $A \to C \to B$ can be derived from Γ. Then, we apply modus ponens two times to the axiom

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

and derive $A \to B$, as desired.

Suppose that $B = \forall x C(x)$ is obtained by generalization as an immediate consequence of $C(x)$, varying the variable x. By assumption, x also does not occur freely in A. By induction hypothesis, we have a derivation $\Gamma \vdash A \to C(x)$. Then we can derive $\forall x(A \to C(x))$ by generalization. By axiom 5, we have

$$\forall x(A \to C(x)) \to A \to \forall x C(x).$$

Then we can derive $A \to \forall x C(x) = A \to B$ by modus ponens, as desired.

Remark 1 The assumption on the free variables in A is clearly necessary, because the generalization rule does not imply that

$$A(x) \to \forall x A(x)$$

is provable. In fact, the formula is not logically valid, it is not hard to give assignments where it is false (e.g. $A(x) :\leftrightarrow x$ is a prime, $x := 3$).

We say that two formulas A and B are logically equivalent, $A \sim B$, iff $\vdash A \leftrightarrow B$.

Exercise 3 Prove that $\sim$ is an equivalence relation.

Theorem 14 (Replacement theorem) *Let C_A be a formula with a subformula A, and let C_B be the formula obtained when A is replaced by B in C_A.*

If $A \sim B$, then $C_A \sim C_B$.

Proof: By induction on the length of the context, we can reduce to a finite number of statements. If the innermost context symbol is $\wedge$, $\vee$, or $\exists$, we may replace both sides by the right hand sides in axioms 9, 10, 13, and use transitivity. Similarily for the context symbol $\leftrightarrow$, because of the theorem

$$(A \leftrightarrow B) \leftrightarrow (A \to B) \wedge (B \leftrightarrow A)$$

(left as an exercise). Thus, it suffices to show the following:

1. $A \leftrightarrow B, C \leftrightarrow D \vdash A \to C \leftrightarrow B \to D$
2. $A \leftrightarrow B \vdash \neg A \leftrightarrow \neg B$
3. $A(x) \leftrightarrow B(x) \vdash \forall x A(x) \leftrightarrow \forall x B(x)$

The first 2 are left as exercises. We show statement 3. As a first step, we show that

$$A(x) \to B(x), \forall x A(x) \vdash \forall x B(x).$$

Indeed, here is a derivation.

1. $\forall x A(x) \to A(x)$	axiom 4
2. $\forall x A(x)$	hypothesis 2
3. $A(x)$	modus ponens from 1 and 2
4. $A(x) \to B(x)$	hypothesis 1
5. $B(x)$	modus ponens from 4 and 3
6. $\forall x B(x)$	generalization of 5

Obviously, x is not free in $\forall x A(x)$. By the deduction theorem, we can derive $\forall x A(x) \to \forall x B(x)$ from $A(x) \to B(x)$.

Under the hypothesis $A(x) \leftrightarrow B(x)$, we may derive $A(x) \to B(x)$ by axiom 7 and $B(x) \to A(x)$ by axiom 8. Then we can prove, as above, the implications $\forall x A(x) \to \forall x B(x)$ and $\forall x B(x) \to \forall x A(x)$. Using axiom 6 and applying modus ponens twice, we obtain the desired $\forall x A(x) \leftrightarrow \forall x B(x)$.

A structure for a first order theory is a non-empty set S, together with an n-ary predicate $\mathbf{P}$ for each n-ary predicate symbol and an n-ary function $\mathbf{f} : \mathbf{S^n} \to \mathbf{S}$ for each n-ary function symbol. An assignment is a map from the set of variables to S. An interpretation is a structure with an assignment.

For each term, we can define its value, an element in S, as in the equational theory.

For each formula, we specify when it is true under the interpretation, in the following way.

- A predicate symbol is true iff the corresponding predicate holds for the values of the argument terms.
- $A \wedge B$ is true iff A is true and B is true.
- $A \vee B$ is true iff A is true or B is true.
- $A \to B$ is true iff A is not true or B is true.
- $\neg A$ is true iff A is not true.
- $\perp$ is not true.
- $\forall x A(x)$ is true iff $A(y)$ is true for all interpretations $\mathcal{I}'$ extending $\mathcal{I}$, where y is a new variable y (which may be assigned any value of S).
- $\exists x A(x)$ is true iff $A(y)$ is true for one interpretation $\mathcal{I}'$ extending $\mathcal{I}$, where y is a new variable y.

It is obvious that relative consistency holds: a provable formula is true in any interpretation (it suffices to check that the axioms are true, and that the inference rules preserve truth). As a consequence, we obtain consistency of first order logic (without additional axioms).

Theorem 15 (Consistency) $\perp$ *is not a theorem.*

Proof: In any interpretation, $\perp$ is not true.

Remark 2 The above proof is not finitary because it uses set theory. It is possible to give a finitary proof, all we have to do is to replace "S is a set" by "S is a finite set of integers". Finiteness is necessary in order to give the definition of truth values of quantified formulas a finitary meaning.

Remark 3 We do not allow empty models, because in these, some axioms and inference rules would not be fulfilled (which?).

Corollary 2 *A necessary condition for being a theorem is that a formula is true under all of its interpretations.*

Exercise 4 Prove that the formula $P(x) \to \forall x P(x)$ is not a theorem.

5 The Completeness Theorem

The goal of this section is to prove semantical completeness of predicate logic. In other words, we want to show that if a formula is true in any interpretation, then it is provable.

A more general result, which also gives some information about sets of formulae, will be useful later.

A set Γ of formulas is called consistent iff there is no derivation of $\perp$ from Γ where all free variables occuring in Γ are held constant. This restriction is important, otherwise $\{P(x), \neg P(y)\}$ would be inconsistent.

Now, we observe that semantical completeness follows from the statement that consistency implies satisfiability.

Proof: Let A be a formula which is true under all interpretations. Then the set $\{\neg A\}$ is not satisfiable. If the second theorem is true, then $\{\neg A\}$ is inconsistent. Thus, there is a derivation $\neg A \vdash \perp$ helding the free variables in A constant. By the deduction theorem, $\neg A \to \perp$ is a theorem. But A is a consequence of $\neg A \to \perp$, hence A is a theorem.

Remark 4 By semantical correctness, it is clear that satisfiability implies consistency: if Γ has a model, then all theorems derivable from Γ are true in this model. But $\perp$ is false in any interpretation, hence $\perp$ is not a consequence of Γ.

A set Γ is called Henkin iff for any formula of the form $\exists x A(x)$, there exists a term t such that Γ contains the formula $\exists x A(x) \to A(t)$.

A set Γ is called maximally consistent iff it is consistent and it is not contained properly in a consistent set.

Theorem 16 (Henkin) *If Γ is Henkin and maximally consistent, then Γ is satisfiable.*

Proof: The model is constructed as follows. H is the set of all terms. For any function symbol f, the function $\mathbf{f} : H^n \to H$ is defined by $\mathbf{f}(t_1, \ldots, t_n) := f(t_1, \ldots, t_n)$. For any predicate symbol P, we define $\mathbf{P}(t_1, \ldots, t_n)$ to be true iff $P(t_1, \ldots, t_n) \in \Gamma$. The assignment I maps any variable to itself.

The remaining part of the proof shows that (H, I) is indeed a model. First, we note that if Γ is maximally consistent and $\Gamma \vdash A$, then we have already $A \in \Gamma$. In a first step, we restrict to a language with only atoms, negation, conjunction, and existential quantifiers. We prove by induction over the number of logical connectives and quantifiers the following claim: any such formula is in Γ iff it is satisfied by the interpretation (H, I). We have to distinguish the following cases.

- A is an atom (base case). Obvious by the definition of H.

- A is a negation $\neg B$. If $A \in \Gamma$, then $B \notin \Gamma$ because Γ is consistent. By induction hypothesis, B is false under the interpretation (H, I). Thus, A is true under the interpretation.

  Conversely, if A is true in the interpretation, then B is false. By the induction hypothesis, $B \notin \Gamma$. Since Γ is maximally consistent, we have $\Gamma, B \vdash \perp$ with a derivation keeping all free variables constant. By the deduction theorem, it follows $\Gamma \vdash B \to \perp$ which implies $\Gamma \vdash \neg B$. Thus $A \in \Gamma$.

- A is a conjunction $B \wedge C$. If $A \in \Gamma$, then $B, C \in \Gamma$ because they are consequences of A. By induction hypothesis, both B and C are true in the interpretation, thus A is true.

  Conversely, if A is true, then both B and C are true. By induction hypothesis, $B, C \in \Gamma$. But $B, C \vdash A$, hence $A \in \Gamma$.

- A is an existential quantification $\exists x B(x)$. If A is true in the interpretation, then there exists a term $t \in H$ such that $B(t)$ is true. By induction, we have $B(t) \in \Gamma$. But $B(t) \vdash \exists x B(x)$, hence $A \in \Gamma$.

  Conversely, let A be in Γ. Since Γ is Henkin, there is a term t such that $A \to B(t) \in \Gamma$. Therefore $B(t) \in \Gamma$. By induction, $B(t)$ is true. Therefore A is true.

If A is a general formula, then there is a formula B which is equivalent to A which has only atoms, negation, conjunction, and existential quantifiers. Therefore A is true iff B is true iff $B \in \Gamma$ iff $A \in \Gamma$.

In the remaining part of the proof, we make the following reasonable assumptions.

- The set of all formulas is countable.
- There are infinitely many variables which are not free in Γ.

Remark 5 The first assumption can be weakened to the assumption that the set of all formulas can be well-ordered. The second assumption is essential. If it does not hold, then a set might be consistent by the unintended reason that there are not enough variables for deriving $\perp$. Such a consistent set will not have a model.

Theorem 17 *Let Γ be a consistent set of formulas. Then there is a set Γ' which contains Γ and is consistent and is Henkin.*

Proof: Let $A_1(x_1), A_2(x_2), \ldots$ be an enumeration of all formulas with a free variable x_n (the variables x_n are not necessarily distinct). We define recursively

$$\Gamma_n := \Gamma_{n-1} \cup \{\exists x_n A_n(x_n) \to A(y_n)\},$$

where y_n is a variable not free in Γ_n. We show that Γ_n is consistent, by induction on n.

Assume indirectly that Γ_n is inconsistent. $B_n := \exists x_n A_n(x_n) \to A(y_n)$. Then we have a derivation $\Gamma_n \vdash \bot$ keeping all free variables in Γ_n constant. By the deduction theorem, we have a derivation $\Gamma_{n-1} \vdash B_n \to \bot$, where $B_n := \exists x_n A_n(x_n) \to A_n(y_n)$. The formula $B_n \to \bot$ is logically equivalent to $\exists x_n A_n(x_n) \wedge (A(y_n) \to \bot)$, therefore we have derivations $\Gamma_{n-1} \vdash \exists x A_n(x_n)$ and $\Gamma_{n-1} \vdash A_n(y_n) \to \bot$. By the third inference rule, we have $A_n(y_n) \to \bot \vdash \exists y_n A_n(y_n) \to \bot$, by a derivation that varies y_n. Now, $\exists x_n A_n(x_n)$ is logically equivalent to $\exists y_n A_n(y_n)$, and therefore we can derive $\Gamma_{n-1} \vdash \bot$ with a derivation that varies y_n. But y_n is not a free variable of Γ_{n-1}, hence Γ_{n-1} is inconsistent. This is a contradiction.

Let $\Gamma' := \cup_n \Gamma_n$. If Γ' were inconsistent, then there would be derivation of $\bot$ with all free variables in Γ' held constant. In this derivation, only finitely many formulas are used. All these finitely many formulas must be contained in some Γ_n. Therefore, Γ_n is inconsistent, but this contradicts the above. Hence Γ' is consistent.

Theorem 18 *Let Γ be consistent. Then there exists a maximally consistent set Γ' containing Γ.*

Proof: Let $A_1, A_2, \ldots$ be an enumeration of the set of all formulas. Define Γ_n recursively by $\Gamma_0 := \Gamma$ and

$$\Gamma_n := \begin{cases} \Gamma_{n-1} \cup \{A_n\} & \text{if } \Gamma_{n-1} \cup \{A_n\} \text{ is consistent,} \\ \Gamma_{n-1} & \text{otherwise.} \end{cases}$$

Then Γ_n is consistent for all n, by induction on n.

Let $\Gamma' := \cup_n \Gamma_n$. If Γ' were inconsistent, then there would be derivation of $\bot$ with all free variables in Γ' held constant. In this derivation, only finitely many formulas are used. All these finitely many formulas must be contained in some Γ_n. Therefore, Γ_n is inconsistent, but this contradicts the above. Hence Γ' is consistent.

Let A_n be a formula not contained in Γ'. Then $A_n \notin \Gamma_n$, and therefore $\Gamma_{n-1} \cup \{A_n\}$ is inconsistent by the recursive definition of Γ_n. Hence $\Gamma' \cup \{A_n\}$ is inconsistent also. Since A_n was chosen arbitrarily, this shows that Γ' is maximally consistent.

Theorem 19 (Completeness) *Any consistent set is satisfiable.*

Proof: Let Γ be a consistent set. By theorem 17, there is a consistent superset Γ' which is Henkin. By theorem 18, there exists a superset Γ'' of Γ' which is maximal consistent. Any superset of a Henkin set is again Henkin, hence Γ'' is also Henkin. By Henkin's theorem, Γ'' is satisfiable. Since Γ is a subset of Γ'', it is satisfiable too.

Note that the Herbrand universe is countable, so we have in fact proved Löwenheim's theorem: any consistent set with a countable set of formulas has a countable model.

Remark 6 The completeness theorem does not hold constructively. The weak point is the use of the excluded middle in the recursive definition of Γ_n in the proof of theorem 18.

Here is a corollary of the completeness theorem which turns out to have some interesting applications in the next chapter.

Theorem 20 (Compactness) *A set Γ is satisfiable iff every finite subset of Γ is satisfiable.*

Proof: If we replace "satisfiable" by "consistent", then the statement is obviously true, because any derivation of $\perp$ uses only finitely many formulas. But by correctness and completeness, satisfiability and consistency are equivalent.

6 First order theories

In this section, we discuss theories that are derived from predicate calculus by adding additional axioms. We will also prove undecidabilty of predicate logic from the undecidability of a special first order theory.

Predicate logic is expressive enough to formalize various theories within it. When doing so, the function symbols symbolize functions from the informal theory, and the predicate symbols symbolize predicates from the informal theory. The number of function symbols and predicate symbols may be finite or infinite. Usually, one has additional axioms. The set of axioms may be infinite, but it must be decidable (this is needed to be able to check proofs).

Formally, a first order theory is defined by a signature (a set of function symbols and predicate symbols together with arity information) and a recursively enumerable set Γ of formulas which is closed under logical consequence. The set Γ is the set of all true formulas in the theory.

Given a first order theory, a model is a structure such that all formulas in Γ are true under all assignments.

A subset Γ_0 of Γ is called an axiomatization iff it is recursive and $\Gamma_0 \vdash A$ for every $A \in \Gamma$.

Theorem 21 *Any first order theory has an axiomatization.*

Proof: Let $A_1, A_2, \ldots$ be a recursive enumeration of Γ. For each n, let $B_n := A_n \wedge A_n \wedge A_n \ldots$ (n times). Then we claim that $\Gamma_0 := \{B_1, B_2, \ldots\}$ is an axiomatization.

In order to decide recursively whether a given formula A is in Γ_0, find the largest number m such that A is a conjunction of m identical formulas. This is easy because it can be done on a syntactical level. For each divisor of l of m, enumerate the first m formulas of Γ. Let A' be the l-th formula. Let B be $A' \wedge A' \wedge A' \ldots$ (l times). If A and B are identical, then A is in Γ_0. If the comparison is negative for all l, then A is not.

Obviously, A_n is a consequence of B_n, hence Γ_0 is an axiomatization.

A theory is called consistent if it does not contain $\bot$. By the completeness theorem, a theory is consistent iff it has a model.

A theory Γ is called complete iff it is consistent and for any closed formula A not contained in Γ, the set $\Gamma \cup \{A\}$ is inconsistent.

Exercise 5 Prove the following theorem:

A theorem is complete iff, for any closed formula A, we have either $\vdash A$ or $\vdash \neg A$ (exclusive or).

A theory Γ is called decidable iff the set Γ is recursive. Any complete theory is decidable: in order to recursively decide whether A is in Γ, enumerate Γ.

Eventually, either A or $\neg A$ will appear. In the first case, $A \in \Gamma$, in the second case $A \notin \Gamma$.

One particular predicate is rather common: equality. It has the axioms

$$x = x,$$

$$x_1 = y_1 \rightarrow \ldots \rightarrow x_n = y_n \rightarrow f(x_1, \ldots, x_n) = f(y_1, \ldots, y_n)$$

for any n-ary function symbol,

$$x_1 = y_1 \rightarrow \ldots \rightarrow x_n = y_n \rightarrow P(x_1, \ldots, x_n) \rightarrow P(y_1, \ldots, y_n)$$

for any n-ary predicate symbol.

Exercise 6 Prove symmetry and transitivity.

Here is a first example of a first order theory: the "one point theory".

- predicate symbols: "=".
- no function symbols.
- axioms:

$$x = y$$

Exercise 7 Prove that the one point theory is complete.

Exercise 8 Give a formalization of a two points theory.

For any equational theory, one can also consider the corresponding first order theory with the same signature and axioms. In this way, one obtains the first order theory of semi-groups, groups, rings, commutative rings.

Theorem 22 *The first order theory of groups is undecidable.*

Proof: Let Σ be an alphabet and Γ be a finite set of equations with an undecidable word problem (see the theorem of Post-Markov in section 1). We reduce the word problem to the decision problem in the theory of groups, thereby showing undecidability of the second.

Let P be the conjunction of all equations in Γ. Given an equation A with words built from Σ, let $E(A)$ be the formula $P \rightarrow A$ where the elements in Σ are considered as constants. We claim that the answer to the instance A of the word problem is yes iff $E(A)$ is a theorem in the first order theory of groups. This achieves the required reduction.

If the answer to the instance of the word problem is yes, then there is a derivation in the equational theory. It is obvious that this derivation can be transformed to a derivation $P \vdash A$ in the first order theory helding all variables constant. By the deduction theorem, $E(A)$ is provable.

Conversely, assume that $E(A)$ is proveable. By semantical correctness, $E(A)$ is true in the following interpretation: Let F be the semi-group of all words in Σ (in other words, the free semi-group generated by Σ). For two words s, t built from Σ, define $s \sim t$ modulo the equational theory defined by Γ. Clearly, this is an equivalence relation. Let S be the set of all equivalence classes. Then the semi-group operation is well-defined for classes, hence S is a model for the theory of semi-groups (i.e. S is a semi-group). Assign each variable in Σ to itself. By semantical correctness, $E(A)$ is true in this interpretation. Since P is also true in this interpretation by construction, the conclusion A must also be true. Hence A is true in the equational theory, and therefore the answer to the instance of the word problem is yes.

Theorem 23 (Church) *Predicate logic is undecidable.*

Proof: Suppose indirectly that we had an algorithm $\mathcal{M}$ for deciding predicate logic. Then, we could use it to decide the theory of groups, as follows. Let A be a given formula in the theory of groups. The theory of groups has only finitely many axioms, which we now abbreviate by $A_1, \ldots, A_r$. Then A is a theorem of group theory iff the formula

$$A_1 \to \ldots \to A_r \to A$$

is a theorem of predicate logic. But this can be decided using the algorithm $\mathcal{M}$.

But the theory of groups is undecidable, a contradiction.

The first order theory of fields is obtained by adding the axiom $x \neq 0 \to \exists y(xy = 1)$ to the equational theory of commutative rings.

The first order theory of ordered fields is obtained by adding a new unary predicate P (positive) to the theory of fields, together with the axioms

1. $P(x) \to \neg P(-x)$;
2. $\neg P(0)$;
3. $P(x) \vee P(-x) \vee x = 0$;
4. $P(x) \wedge P(y) \to P(x + y)$;
5. $P(x) \wedge P(y) \to P(xy)$.

This is also a first example of a consistent theory without a finite model.

The first order theory of algebraically closed fields is obtained by adding an axiom

$$\exists x(x^n + y_1 x^{n-1} + \cdots + y_n = 0) \tag{1}$$

for each positive integer n to the theory of fields. Of course, one has to write the powers by expanding them into products, in order to express them in the available language.

Theorem 24 *The first order theory of algebraically closed fields is complete (and consequently decidable).*

Proof: See [23].

The first order theory of real closed fields is obtained by adding an axiom 1 for each odd number n, and by adding the axiom

$$P(x) \to \exists y(yy = x).$$

Theorem 25 *The first order theory of real closed fields is complete (and consequently decidable).*

Proof: See [23].

Exercise 9 Give a formalization of the theory of fields of characteristic zero.

Here is an example of a class of structures which is not the class of models of a first order theory.

Proposition 1 *The class of finite groups is not the class of a first order theory.*

Proof: Suppose indirectly that finite groups can be characterized by a first order theory Γ. Let A_n be the formula

$$\exists(x_1, \ldots, x_n) : (x_1 \neq x_2 \wedge \cdots \wedge x_{n-1} \neq x_n).$$

The groups satisfying A_n are exactly the groups with at least n elements. Let $\Delta := \{A_1, A_2, \ldots\}$.

Any finite subset of $\Gamma \cup \Delta$ is satisfiable by a group of sufficiently large order. By the compactness theorem, it follows that $\Gamma \cup \Delta$ is satisfiable. Hence there is an infinite group satisfying Γ, a contradiction.

7 Elementary Theory of Real Numbers

We introduce a very important first order theory, the theory of real numbers. Methodologically, we assume a large part of classical methematics (especially algebra), because the participants are probably familiar with it.

Let $\mathcal{F}$ be the theory of fields, with predicate symbol "=", function symbols $+, -, 0, \cdot, 1$, ring axioms, and the axiom

$$\neg x = 0 \to \exists y\ xy = 1.$$

The theory $\mathcal{OF}$ (ordered fields) is an extension with additional predicate symbol $<$ and additional axioms

$$\neg x < x, x < y \to y < z \to x < z,$$

$$x < y \vee y < x \vee x = y,$$

$$x < y \to x + z < y + z,$$

$$0 < z \to x < y \to xz < yz.$$

Obviously, one can define the predicates $>, \leq, \geq, \neq$.

Remark 7 About defining new predicates: it is possible to interpretate the new predicates as metatheoretical abbreviations, e.g. $t_1 \leq t_2$ is nothing but an abbreviation of $t_1 = t_2 \vee t_1 < t_2$. Alternatively, we can extend the formal system by adding the new predicate symbol and add the new axiom

$$x_1 \leq t_2 \leftrightarrow (x1 = x_2 \vee x_1 < x_2).$$

One can show formally that the original system and the extended system are equivalent, in the sense that derivations can be translated in both directions.

A very familiar model for the theory $\mathcal{OF}$ is $\mathbf{Q}$.

By composition of the available function symbols, we get all multivariate polynomials with integral coefficients. We "identify" polynomials which provably coincide for all arguments, e.g. $(x + x) \cdot x = x \cdot x + x \cdot x + 0 \cdot x$. Doing so, we can bring polynomials into normal form, and we can define the degree of a polynomial with respect to a variable. By definition, we say that the zero polynomial has degree $-\infty$.

If $a_0, \ldots, a_n$ are variables, then the polynomial

$$\Gamma_n := a_0 + \ldots + a_n x^n$$

is called a *generic polynomial* of degree n (obviously, generic polynomials are unique up to renaming of variables).

The theory $\mathcal{R}$ of real numbers is an extension of $\mathcal{OF}$ by an infinite number of axioms, one for each natural number n:

$$\Gamma_n(x) < 0 \to \Gamma_n(y) > 0 \to \exists z(x < z < y \wedge \Gamma_n(z) = 0)$$

(called intermediate value axioms).

Of course, the intended model is the set of real numbers. Note, however, that this model is not a model in the sense as defined in the previous section, because real numbers cannot be finitely represented.

Here is a non-standard model which consists of finitely described objects: the set of all real algebraic numbers.

Exercise 10 Prove

$$x > 0 \to \exists y(y^2 = x).$$

Theorem 26 *Let* $\Gamma_n := a_0 + \ldots + a_n x^n$, *$n$ odd. Then*

$$\vdash a_n \neq 0 \to \exists x \Gamma_n(x) = 0.$$

Proof: Assume $a_n \neq 0$. For $i = 0, \ldots, n-1$, we know $\exists b_i\ a_n b_i = a_i$. For these b_i,

$$\exists b(b > -nb_0 \wedge \ldots \wedge b > -nb_{n-1} \wedge b > 1),$$

since $<$ is a total order. For this b, we have $b^n > -nb_i x^i$, $i = 0, \ldots, n-1$. Therefore

$$b_0 + \ldots + b_{n-1}b^{n-1} + b^n > 0.$$

Similarily, we construct an a such that

$$b_0 + \ldots + b_{n-1}a^{n-1} + a^n < 0.$$

By the intermediate value axiom, the polynomial $b_0 + \ldots + b_{n-1}x^{n-1} + x^n$ has a zero c. But then, we also have

$$a_0 + \ldots + a_n c^n = 0.$$

Some authors give a different axiomatization of $\mathcal{R}$. They omit the intermediate value axioms; instead, they postulate the existence of square roots for positive elements (exercise 10), and the existence of roots of polynomials of odd degree (theorem 26). To show that the two axiomatizations are equivalent, one has to show the intermediate value statement from the exercise 10 and theorem 26.

Theorem 27 *Let K be an ordered field such that any positive element has a square root and any polynomial of odd degree has a root. Then the intermediate value theorem holds in K.*

Proof: See (Jacobson, Basic Algebra I).

We are now able to prove that the intermediate value axioms follow formally from exercise 10 and theorem 26. For any n, let I_n be the universal quantification of the n-th intermediate value statement. The negation $\neg I_n$ together with $\mathcal{OF}$ and exercise 10 and theorem 26 is insatisfiable. By theorem **??**, this set is inconsistent, hence there is a derivation of $\perp$. By the deduction theorem, there is a derivation of $\neg I_n \to \perp$ from the remaining axioms. From this we can formally deduce the intermediate value axioms.

Remark 8 Note that the proof above is not a finitary one, because it uses the infinite models. To obtain a finitary proof, it is necessary to translate the proof of theorem 27 into our formal language.

In one of the next sections, it is shown that $\mathcal{R}$ is complete, hence decidable. We already know that it is consistent, because it has a model. Caution: we do not yet have a finitary proof for consistency!

Actually, completeness will be obtained from the following theorem, which is interesting on its own.

Theorem 28 (Quantifier elimination) *Let A be a formula in $\mathcal{R}$. Then there exists a formula B without quantifiers, such that $A \sim B$ in $\mathcal{R}$.*

In other words, any definable n-ary is equivalent to a Boolean combination of polynomial equations and inequalities. Especially, any unary predicate is a finite union of intervals and points. For instance, the properties of being rational/integral cannot be expressed in the language $\mathcal{R}$.

Completeness follows from the obvious fact that any closed formula without quantifiers is either a theorem or contradictory.

For later use, we mention Dedekind's cuts. This is a scheme of formulas of the form

$$(A(x) \to B(y) \to x \leq y) \to \exists b (A(x) \to B(y) \to x \leq b \leq y)$$

which is well-known from analysis. Since these formulae are true in $\mathbf{R}$, and - as we will see - the system $\mathcal{R}$ is complete, the formulae can be proven from the axioms. Alternatively, one can also show that the Dedekind's cuts are theorems by theorem 28.

8 Formal Euclidean Geometry

In this section, we present maybe the most outstanding result obtained of a long history of axiomatization: decidability of euclidean geometry. Most proofs are only sketched, due to the fact that they are lengthy but not surprising.

Usually, euclidean geometry is formalized using two types, points and lines (or points, lines, and planes). In our formalization, we follow Tarski, who observed that the type "lines" is eliminable.

In addition to equality, we have two predicates. The ternary predicate $Bxyz$ has the following semantic interpretation: the three points x, y, z are collinear and y is between x and z (or coincides with x or z). The 4-ary predicate $xy{\cong}zw$ has the following semantic interpretation: the distance between x and y is equal to the distance of z and w. We define a new ternary predicate by

$$Cabc :\leftrightarrow Babc \vee Bbca \vee Bcab,$$

which has the interpretation that a, b, c are collinear.

We have 10 axioms and one axiom scheme (which gives infinitely many axioms).

1. $ab \cong ba$
2. $ab \cong pq \to ab \cong rs \to pq \cong rs$
3. $ab \cong cc \to a = b$
4. $\exists x(Bqax \wedge ax \cong bc)$
5. $a{\neq}b \to Babc \to Ba'b'c' \to ab{\cong}a'b' \to bc{\cong}b'c' \to ad{\cong}a'd' \to bd{\cong}b'd' \to cd{\cong}c'd'$
6. $Baba \to a = b$
7. $Bapc \to Bbqc \to \exists x(Bbxp \wedge Bqxa)$ (Pasch's axiom)
8. $\exists a \exists b \exists c \neg Cabc$ (lower dimension axiom)
9. $p{\neq}q \to ap \cong aq \to bp \cong bq \to cp \cong cq \to Cabc$ (upper dimension axiom)
10. $Badt \to Bbdc \to a{\neq}d \to \exists x \exists y(Babx \wedge Bacy \wedge Bxty)$

(10) is equivalent to Euclid's axiom of parallels. The axiom scheme is

$$\exists a \forall x \forall y(A(x){\to}B(y){\to}Baxy) \to \exists b \forall x \forall y(A(x){\to}B(y){\to}Bxby).$$

This is a geometric formulation of Dedekind's axiom of cuts.

Obviously, the extended model is the Euclidean plan, with points as objects. Lines are superfluous because any assertion about lines can be expressed by the betweenness relation.

One can give an axiomatization for the Euclidean 3-space (or higher dimensional Euclidean spaces) by replacing axioms 8 and 9 by other axioms (see [10]).

Here are some easy theorems.

Theorem 29 *The following formulas are theorems.*

1. $ab \cong ab$
2. $ab \cong cd \to cd \cong ab$
3. $ab \cong cd \to cd \cong ef \to ab \cong ef$
4. $aa \cong bb$
5. $Babc \to Ba'b'c' \to ab{\cong}a'b' \to bc{\cong}b'c' \to ac{\cong}a'c'$
6. $q{\neq}a \to \exists!x(Bqax{\wedge}ax{\cong}bc)$
7. $Babb$
8. $Babc \to Bcba$
9. $Babc \to Bbac \to a = b$
10. $\exists x \exists y\ x{\neq}y$

Proof: We prove the first, and the eigth assuming the seventh.

1. $ba \cong ab \to ba \cong ab \to ab \cong ab$ axiom 2
2. $ba \cong ab$ axiom 1
3. $ba \cong ab \to ab \cong ab$ modus ponens from 1 and 2
4. $ab \cong ab$ modus ponens from 3 and 2

To show the eigth, we assume $Babc$ and $Bbcc$ (by the seventh) and derive $Bcba$.

1. $Babc \to Bbcc \to \exists x(Bbxb \wedge Bcxa)$ axiom 7
2. $\exists x(Bbxb \wedge Bcxa)$ modus ponens two times

By axiom 6, such an x is unique and it must be b, hence $Bcba$.

Exercise 11 Prove three of the above formal theorems (not the first and the eighth).

Here is a theorem which is geometrically more interesting.

Theorem 30 *The following formula is a theorem.*

$$Babc \to Bdbe \to ab{\cong}db \to bc{\cong}be \to ae{\cong}cd$$

Proof: We distinguish the cases $a{\neq}b$ and $a = b$.

Assume $a = b$. Then $d = b$ by the assumption $ab{\cong}db$ and axiom 3 (using the symmetry facts on $\cong$ which have been proved in the above exercise). Then the assumption $bc{\cong}be$ implies the conclusion $ae{\cong}cd$.

Now, assume $a{\neq}b$. By the specialization $a' = d$, $b' = b$, $c' = e$, $d' = a$, from axiom 5, we obtain

$$a{\neq}b \to Babc \to Bdbe \to ab{\cong}db \to bc{\cong}be \to ad{\cong}da \to bd{\cong}ba \to cd{\cong}ea.$$

All assumptions of the axioms are fulfilled in our case, hence also the conclusion $cd{\cong}ea$, which implies $ae{\cong}cd$.

To be more flexible, we enrich our language a little bit. Similar to the definition of new predicates, we may also define new functions. Let $\exists!$ be a shorthand for unique existence, i.e. $\exists! x A(x)$ stands for

$$\exists x A(x) \wedge (A(x) {\to} A(y) {\to} x = y).$$

Suppose that we have proven a theorem

$$\exists! y A(x_1, \ldots, x_n, y),$$

where A has no other free variables than $x_1, \ldots, x_n, y$. Then we may extend the language by adding a new n-ary function symbol f and the new axiom

$$A(x_1, \ldots, x_n, f(x_1, \ldots, x_n)).$$

One can show that this extension is an equivalence, in the sense that derivations can be translated in both directions. More precisely, we translate formulas of the kind $P(s_1, \ldots, s_{k-1}, f(t_1, \ldots, t_n), s_{k+1}, \ldots, s_m)$ into the formula

$$\exists y (P(s_1, \ldots, s_{k-1}, y, s_{k+1}, \ldots, s_m) \wedge A(t_1, \ldots, t_n, y)),$$

where y is a variable not occuring in the terms involved.

In this sense, we define a new binary function s (point reflection) by

$$Bbas(a, b) \wedge as(a, b){\cong}ab.$$

We also write $s_a(b)$ instead of $s(a, b)$. One can show that s_a is an involutory map preserving distance and between-ness:

$$s_a(s_a(b)) = b$$

$$Bbcd \leftrightarrow Bs_a(b)s_a(c)s_a(d)$$

$$bc{\cong}de \leftrightarrow s_a(b)s_a(c){\cong}s_a(d)s_a(e)$$

Then, we define a new ternary predicate (right angle) by

$$Rabc :\leftrightarrow ac \cong s_b(a)c.$$

One can show that $\exists! x(Baxb \wedge ax{\cong}xb)$. We use this to define a binary function m (midpoint) by

$$Bam(a,b)b \wedge am(a,b){\cong}m(a,b)b.$$

We define a 4-ary predicate (parallel lines) by

$$ab||cd :\leftrightarrow \exists(x,y)(Rbax \wedge Raby \wedge Ccxd \wedge Ccyd \wedge ax{\cong}by).$$

Exercise 12 Prove

$$Babc \to Bade \to ab{\cong}ad \to bc{\cong}de \to be{\cong}cd$$

Our next goal is to show that our formalization $\mathcal{G}$ of Euclidean geometry is equivalent to $\mathcal{R}$, the theory of real closed fields. This implies that $\mathcal{G}$ is decidable.

The idea goes back to Descartes. He had in mind an algorithm with the following specification.

Input: A formula A of $\mathcal{G}$.

Output: A formula $B = F(A)$ of $\mathcal{R}$, which is a theorem iff A is a theorem.

To describe Descartes' algorithm, we describe how to deal with the formulas $a = b$, $Babc$, and $ab{\cong}cd$. These will be translated into certain formulas $E(a_1, a_2, b_1, b_2)$, $B(a_1, a_2, b_1, b_2, c_1, c_2)$, and $D(a_1, a_2, b_1, b_2, c_1, c_2, d_1, d_2)$. The logical connectives $\wedge, \vee, neg, \to, \perp$ are translated 1:1. The quantifiers are doubled, i.e. $\exists x$ translates to $\exists x_1 \exists x_2$, and $\forall x$ translates to $\forall x_1 \forall x_2$.

$$E \text{ is } a_1 = b_1 \wedge a_2 = b_2,$$

$$B \text{ is } a_1(b_2 - c_2) + b_1(c_2 - a_2) + c_1(a_2 - b_2) = 0 \wedge (b_2 - c_2)(a_2 - b_2){\geq}0 \wedge (b_1 - c_1)(a_1 - b_1){\geq}0,$$

$$D \text{ is } (a_1 - b_1)^2 + (a_2 - b_2)^2 = (c_1 - d_1)^2 + (c_2 - d_2)^2.$$

Exercise 13 Give a translation of the predicates R and $||$ (without formal proof).

Theorem 31 *If $A_1, \ldots, A_n \vdash C$ in $\mathcal{G}$, then $F(A_1), \ldots, F(A_n) \vdash F(C)$ in $\mathcal{R}$.*

Proof: (sketch) It suffices to prove the statement for all material axioms, because we did not do any changes in the logical structur (except for the duplication of quantifiers, but here all axioms/inference rules carry over).

The first 10 axioms can be checked automatically, this is cumbersome but easy.

Now, let $A(x)$ and $B(y)$ be formulas. Let D be the axiom arising from A and B by the Dedekind scheme. Let $A(x_1, x_2)$ and $B(x_1, x_2)$ be the real translations of A and B. Using some case distinction, we can derive $F(D)$ from the formula

$$(A'(x) \to B'(y) \to x \leq y) \to \exists b (A'(x) \to B'(y) \to x \leq b \leq y)$$

(Dedekind cuts). It can be shown that this formula is a theorem.

Corollary 3 *$\mathcal{G}$ is consistent.*

Proof: If $\vdash \perp$ in $\mathcal{G}$, then $\vdash F(\perp) = \perp$ in $\mathcal{R}$, but we know that $\mathcal{R}$ is consistent.

If we have a model $(R, +, -, 0, \cdot, 1, <)$ of the theory $\mathcal{R}$, then we can construct a model $(G, B, \cong)$ as follows.

- Set $G := R^2$.
- Set B and $\cong$ as in the translation above.

It is obvious that the truth value of a closed formula A in the interpretation $(G, B, \cong)$ is equal to the truth value of the formula $F(A)$ in the interpretation $(R, +, -, 0, \cdot, 1, <)$.

The following theorem states that every model can be constructed in this way.

Theorem 32 *Let $(G, B, \cong)$ be a model of $\mathcal{G}$. Then there exists a model $(R, +, -, 0, \cdot, 1, <)$ of $\mathcal{R}$, such that $(G, B, \cong)$ is isomorphic to $(R^2, B', \cong')$, where B' and $\cong'$ are defined as above.*

Proof: In $\mathcal{G}$, one can prove the formula

$$\exists (o, e_1, e_2)(o \neq e_1 \wedge oe_1 \cong oe_2 \wedge Re_1oe_2).$$

Since G is a model of $\mathcal{G}$, there exist three elements in $o, e_1, e_2 \in G$ such that $o \neq e_1$, $oe_1 \cong oe_2$, and Re_1oe_2.

We set R as the subset of G of all elements x for which Coe_1x is true (the first coordinate axis).

We set $0 := o$ and $1 := e_1$. These points are in $\mathcal{R}$ because one can derive in $\mathcal{G}$

$$o \neq e_1, oe_1 \cong oe_2, Re_1oe_2 \vdash Coe_1o, Coe_1e_1.$$

For $x, y \in R$, we set $x + y := s_o(m(x, y))$. In order to show that these points are in R, one has to prove in $\mathcal{G}$

$$o \neq e_1, oe_1 \cong oe_2, Re_1oe_2, Coe_1x, Coe_1y \vdash Coe_1s_o(m(x, y)).$$

For $x \in R$, we define $-x := s_o(x)$ (again we have to show that this is in R).

For $x, y \in R$, we define $x < y$ to be true iff $\neg B(a - b)oe_1$ is true.

The definition of multiplication is a bit tricky. First, we prove in $\mathcal{G}$ the consequence

$$o \neq e_1, oe_1 \cong oe_2, Re_1oe_2, Coe_1x, Coe_1y \vdash \exists! z Coe_1z \wedge (\exists w(Coe_2w \wedge e_1e_2||yw \wedge e_2x||wz)).$$

Then, for given x, y in R, we know that there exists a unique z for which the above existential formula is true. Moreover, this z is in R because Coe_1z is true. Thus, we can define $x \cdot y$ to be this z.

The next step is to show that the structure just defined satisfies all axioms of $\mathcal{R}$. We will not do this here.

It remains to construct the isomorphism between G and R^2. This is done as follows.

For $a \in G$, define $c_1(a) \in R$ as the unique element such that $(Roc_1(a)a \wedge Coe_1c_1(a)$ is true (again one can show uniqueness and existence in $\mathcal{G}$). Define $c_2(a) \in R$ as the unique element such that

$$(Roc_2(a)a \wedge \exists u(Roua \wedge Coe_2u \wedge e_1e_2||uc_2(a))$$

is true. Then $a \mapsto (c_1(a), c_2(a))$ defines a map from G to R^2. One can show that this map is an isomorphism, and finally that the this isomorphism takes the relations B and $\cong$ to the relations B' and $\cong'$ as above.

Corollary 4 *$\mathcal{G}$ is complete.*

Proof: Assume indirectly that A be a closed formula of $\mathcal{G}$ such that A and $\neg A$ are both not provable. Then $\mathcal{G} \cup \mathcal{A}$ is consistent. By the completeness theorem, there is a model $(G, B, \cong)$ satisfying A. By the above theorem, we can assume that $G = R^2$ for a real closed field R. Therefore R is a model for $F(A)$. Because of semantical correctness, $\mathcal{R} \cup \mathcal{F}(\mathcal{A})$ is consistent. Because $\mathcal{R}$ is complete, $F(A)$ is a theorem of $\mathcal{R}$.

Likewise, $\mathcal{G} \cup \neg\mathcal{A}$ is consistent, so there is a model G' satisfying $\neg A$, we may assume $G' = (R')^2$, and R' is a model for $F(\neg A) = \neg F(A)$, so $\mathcal{R} \cup \neg\mathcal{F}(\mathcal{A})$ is consistent, so $\neg F(A)$ is a theorem of $\mathcal{R}$. But then $\perp$ is a theorem of $\mathcal{R}$, which contradicts consistency of $\mathcal{R}$.

9 Formal Number Theory

A theory of natural numbers is fundamental for many parts of mathematics.

The formal system of number theory (also called the theory of arithmetic) has one constant 0, one unary function symbol $'$ (successor) and two binary functions $+$ and $\cdot$. There are various axiomatizations. A common one is Peano's axioms:

1. $a' = b' \to a = b$;
2. $\neg a' = 0$;
3. $a + 0 = a$;
4. $a + b' = (a + b)'$;
5. $a \cdot 0 = 0$;
6. $a \cdot b' = a \cdot b + a$;
7. $A(0) \to \forall x(A(x) \to A(x \leftarrow x')) \to A(x)$, where A is a formula.

In this section, we will not be very much with a specific axiomatization. Instead, we will show that there exists no complete axiomatization.

The standard model is the set $\mathbb{N}$, which can be constructed as the terms $0, 1 := 0', 2 := 0'', \ldots$. The natural numbers constructed this way are also called numerals. Addition and multiplication for numerals are defined as usual (there is a unique way which is consistent with the Peano axioms).

We will need meta-variables for numerals, which must not be confused with the variables in the formal system. Thus, we make the following convention:

- The letters a, b, c, d, e, x, y, z are reserved for variables.
- The letters i, j, k, l, m, n are reserved for meta-variables ranging over numerals.
- The function symbols $+$ and $\cdot$ are used as a language element and as names for function on numerals. One has to look at the arguments to tell. We will not use $'$ as a name for a function on numerals.

We say that an n-ary predicate P of numerals is arithmetic iff there exists a formula $A(x_1, \ldots, x_n)$ such that for every n-tuple $(i_1, \ldots, i_n)$ of numerals, $P(i_1, \ldots, i_n)$ is equivalent to the statement that $A(i_1, \ldots, i_n)$ is a true in the standard model.

A partial n-ary function f from numerals to numerals is called arithmetic iff the $n + 1$-ary predicate $f(i_1, \ldots, i_n) = j$ (the graph of f) is arithmetic.

Theorem 33 *Let $f_1, \ldots, f_m$ be n-ary arithmetic functions of numerals, and let P be an P be an arithmetic predicate. Then the predicate $P(f_1(i_1, \ldots, i_n), \ldots, f_m(i_1, \ldots, i_n))$ is arithmetic.*

Proof: Let f be a unary arithmetic function, expressed by the formulas $F(x, y)$. Let P be a unary arithmetic predicate, expressed by the formula $A(x)$. Then $P(f(i))$ holds if and only if there exists a numeral j such that $F(i, j)$ and $A(j)$ are both true in the standard model. Since the standard model contains exactly the numerals, this is equivalent to $\exists z(F(i, z) \wedge A(z))$ being true in the standard model. Thus, the compound predicate is expressed by the formula $\exists z(F(x, z) \wedge A(z))$.

The general case with several arguments is similar.

Corollary 5 *The composition of arithmetic functions is arithmetic.*

Proof: Apply theorem 33 to the graph of the outer function.

An essential feature of formal number theory is that one can construct tuples (i.e. finite sequences).

Theorem 34 (Gödel's Beta-function) *There is an arithmetic ternary function β on numerals, such that for any finite sequence $i_0, \ldots, i_n$ of numerals, there exist numerals j, k, such that $\beta(j, k, m) = i_m$.*

Proof: The definition of β is quite simple, namely we take

$$\beta(j, k, m) := rem(j, 1 + (m + 1)k).$$

The hard part is to construct j and k for a given sequence. Let $i_0, \ldots, i_n$ be a sequence. Let l be the maximum of $n, i_0, \ldots, i_n$, and take $k := l!$. Then the numerals $1 + k, 1 + 2k, \ldots, 1 + (n + 1)k$ are pairwise relatively prime. By the Chinese remainder theorem, there exists a numeral j such that

$$rem(j, 1 + (m + 1)k) = i_m$$

for $m = 0, \ldots, n$.

In order to see that β is arithmetic, it suffices to notice that $\beta(i, j, k) = l$ is equivalent to the truth of

$$\exists x((m' \cdot k)' \cdot x + l = j) \wedge \exists y(l + y' = (m' \cdot k)')$$

in the standard model.

Remark 9 *In the following, we write $B(x, y, z, w)$ for the formula expressing $\beta(x, y, z) = w$.*

Recall the definition of recursive functions (see [32] or [24]), constructed from the zero function, the successor function, and the projection functions, by composition, primitive recursion, and minimalization. The partial recursive functions are defined analoguously, except that minimalization is defined in the following way: if $g(i,j)$ is a partial recursive function, then the minimalization $\mu(g)$ is defined by $\mu(g)(j)$ is the smallest i such that $g(i,j)=0$ and $g(i',j)$ is defined for all $i'<i$.

Theorem 35 *Any partial recursive function f is arithmetic.*

Proof: Obviously, the constant zero function, the successor, and the projection functions are arithmetic. Composition was already proven to preserve arithmetic-ness.

Primitive recursion: let l be a numeral, and let h be a binary function expressed by the formula $H(x_1,x_2,y)$. Then we claim that the primitive recursion f—defined by $f(0):=l$, $f(n+1)=h(i,f(i))$—is expressed by the formula

$$F(x,y) := \exists(c,d)$$

$$(B(c,d,0,l)\wedge$$

$$\forall z(z<x\to\exists(a,b)(B(c,d,z,a)\wedge B(c,d,z',b)\wedge H(z,a,b))\wedge$$

$$B(c,d,x,y).$$

Suppose that $f(i)=j$. By the theorem of the β-function, there exist numerals m,n, such that $\beta(m,n,k)=f(k)$ for all $k=0,\ldots,i$. Therefore the formulas $B(m,n,k,f(k))$ are true in the standard model. Since $f(0)=l$, the formula $B(m,n,0,l)$ is true. Since $f(k+1)=h(i,f(k))$, the formulas $B(m,n,k,f(k))$, $B(m,n,k',f(k+1))$, and $H(k,f(k),f(k+1))$ are true for all $k<i$. Since $f(i)=j$, the formula $B(m,n,i,j)$ is true. This proves that $F(i,j)$ is true in the standard model.

Conversely, assume that $F(i,j)$ is true in the standard model. Then there exist numerals m,n such that the formulas

$$\begin{aligned}B(m,n,0,l),\\ \forall z(z<i\to\exists(a,b)(B(m,n,z,a)\wedge B(m,n,z',b)\wedge H(z,a,b)),\\ B(m,n,i,j)\end{aligned}$$

are all true. The first implies $\beta(m,n,0)=l$. The second implies that sequence $i_k:=\beta(c,d,k)$, $k=0,\ldots,i-1$, fulfills the recursion $h(k,i_k)=i_{k+1}$. By the definition of f, it follows that $f(k)=i_k$ for $k=0,\ldots,i$. Finally, the third formula being true implies that $i_i=j$. It follows that $f(i)=j$.

The general case with several arguments is treated analogously.

Minimalization: Suppose that g is an binary partial function described by the ternary formula $G(x_1, x_2, y)$. Then we claim that the minimalization f of g—defined by $f(i)$ is the smallest j such that $g(j, i) = 0$, supposed that $g(j', i)$ is defined for all $j' < j$—can be described by the formula

$$F(x, y) := G(y, x, 0) \wedge \forall z(z < y \rightarrow \exists w(G(z, x, w) \wedge w \neq 0)).$$

Let $f(i) = j$. Then $g(i, j) = 0$, hence $G(y, x, 0)$ is true in the standard model. For all $k < j$, there exists $l > 0$ such that $g(k, i) = l$. It follows that $F(i, j)$ is true.

Conversely, suppose that $F(i, j)$ is true in the standard model. Then $g(j, i) = 0$, and for all $k < j$, there exists $l > 0$ such that $g(k, i) = l$. Hence $f(i) = j$ be the definition of minimalization.

The general case with several arguments is treated analogously.

Corollary 6 *For any recursively enumerable subset P of numerals, the property "contained in P" is arithmetic.*

Proof: Let P be a recursively enumerable set of numerals. It is well-known that a set is recursively enumerable iff it is the domain of a partial recursive function. By theorem 35, any such function is arithmetic. Let $F(x, y)$ be the formula expressing a partial recursive function with domain P. Then the property "contained in P" is expressed by $A(x) := \exists y F(x, y)$.

Theorem 36 (Gödel's incompleteness theorem) *Any first order theory with equality, a constant symbol 0, a unary function symbol $'$, and two binary function symbols $+$ and $\cdot$, which is satisfied by the standard model $\mathbb{N}$, is incomplete.*

Proof: Suppose, indirectly, that we have a complete axiomatization. Let Q be a recursively enumerable set of numerals which is not recursive. By corollary 6, the property "contained in Q" is arithmetic. Hence it can be expressed by a formula $A(x)$.

For any given numeral n, we have either $\vdash A(n)$ or $\vdash \neg A(n)$ by completeness. Thus, we can produce algorithmically either a proof of $A(n)$ or a proof of $\neg A(n)$. Thus, we can decide whether n is contained in Q. This contradicts the assumption that Q is not recursive.

In contrast to the incompleteness theorem, consider the theory of Presburger arithmetic: one constant 0, one unary function symbol $'$ (successor) and one binary functions $+$. The axioms are:

1. $a' = b' \rightarrow a = b$;
2. $\neg a' = 0$;

3. $a + 0 = a$;
4. $a + b' = (a + b)'$;
5. $A(0) \to \forall x(A(x) \to A(x \leftarrow x')) \to A(x)$, where A is a formula.

Theorem 37 (Presburger) *The theory of Presburger arithmetic is complete (hence decidable).*

Proof: See [23].

10 Hilbert's 10th problem

In 1900, Hilbert formulated the problem of giving an algorithm to decide whether a given polynomial equation in several variables, $P(x_1, \ldots, x_n) = 0$, has a solution in integers. In 1970, Matijasevic proved that such an algorithm is not possible. In this section, we will give a proof for this result, assuming a certain technical lemma (exponentiation is diophantine).

We say that an n-ary predicate P of numerals is diophantine iff there exists a numeral m and a polynomial $Q \in \mathbb{Z}[x_1, \ldots, x_n, y_1, \ldots, y_m]$, such that $P(i_1, \ldots, i_n)$ is true iff the diophantine equation

$$Q(i_1, \ldots, i_n, y_1, \ldots, y_m) = 0$$

has a solution $(j_1, \ldots, j_m) \in \mathbb{N}^m$.

Proposition 2 *Every diophantine predicate is recursively enumerable.*

Proof: The following program, with input $i_1, \ldots, i_n$,

for all m-tuples $(j_1, \ldots, j_m)$
if $Q(i_1, \ldots, i_n, j_1, \ldots, j_m) = 0$, then exit

has Q as its domain of definition. In the for loop, one has to take care that all possible tuples are enumerated eventually.

The main result of this section will be the converse of this proposition.

It is clear that every diophantine predicate is arithmetic, since it can be expressed by the formula

$$\exists(y_1, \ldots, y_m)(Q_+(x_1, \ldots, x_n, y_1, \ldots, y_m) = Q_-(x_1, \ldots, x_n, y_1, \ldots, y_m)),$$

where Q is written as a difference $Q_+ - Q_-$ with polynomials with coefficients in $\mathbb{N}$.

Lemma 2 *The conjunction and disjunction of diophantine predicates is diophantine.*

Proof: Let

$$P_1(x_1, \ldots, x_n) :\leftrightarrow \exists(y_1, \ldots, y_m)(Q_1(x_1, \ldots, x_n, y_1, \ldots, y_m) = 0),$$

$$P_2(x_1, \ldots, x_n) :\leftrightarrow \exists(z_1, \ldots, z_l)(Q_2(x_1, \ldots, x_n, z_1, \ldots, z_l) = 0),$$

be diophantine predicates (negative coefficients of Q_1 and Q_2 have to be syntactically understood as coefficients on the right hand side of the equation). Then $P_1 \wedge P_2$ can be expressed by

$$\exists(y_1, \ldots, y_m, z_1, \ldots, z_l)(Q_1(x_1, \ldots, y_m)^2 + Q_2(x_1, \ldots, z_l)^2 = 0),$$

and $P_1 \vee P_2$ can be expressed by

$$\exists(y_1, \ldots, y_m, z_1, \ldots, z_l)(Q_1(x_1, \ldots, y_m)Q_2(x_1, \ldots, z_l) = 0).$$

Lemma 3 *Existential quantification of diophantine predicates is diophantine.*

Proof: Let

$$P(x_1, \ldots, x_m) :\leftrightarrow \exists(y_1, \ldots, y_m)(Q(x_1, \ldots, x_n, y_1, \ldots, y_m) = 0)$$

be a diophantine predicate. Then $\exists x_1 P(x_1, \ldots, x_m)$ can be expressed by

$$\exists(y_1, \ldots, y_m, x_1)(Q(x_1, \ldots, x_n, y_1, \ldots, y_m) = 0).$$

We say that an n-ary partial function is diophantine iff its graph (an $n+1$-ary predicate) is diophantine. For example, all polynomial partial functions—being undefined iff the result is negative—are diophantine, because they can be expressed by

$$\exists()(Q_+(x_1, \ldots, x_n) = Q_-(x_1, \ldots, x_n) + y)$$

(zero quantifications). Especially, the functions $+$ and $\cdot$ are diophantine.

Lemma 4 *Let $f_1, \ldots, f_m$ be n-ary diophantine functions, and let P be an P be a diophantine predicate. Then the predicate $P(f_1(i_1, \ldots, i_n), \ldots, f_m(i_1, \ldots, i_n))$ is diophantine.*

Proof: Replace "arithmetic" by "diophantine" in the proof of theorem 33.

Corollary 7 *The composition of diophantine functions is diophantine.*

Proof: Apply lemma 4 to the graph of the outer function.

Exercise 14 Here are some diophantine predicates and functions:

$$x < y \leftrightarrow \exists z(x + z + 1 = y)$$

$$x|y \leftrightarrow \exists z(xz = y)$$

$$div(x, y) = z \leftrightarrow \exists(u, v)((u + v + 1 - y)^2 + (u - x + zy)^2 = 0)$$

$$rem(x, y) = z \leftrightarrow \exists(u, v)((z + u + 1 - y)^2 + (z - x + vy)^2 = 0)$$

Lemma 5 *Gödel's β-function is diophantine.*

Proof: It suffices to observe that the β-function is a composite of the remainder function and polynomial functions.

At this point, we see that we could copy the proof of theorem 35, replacing by "arithmetic" by "diophantine", if we only knew that bounded universal quantification of diophantine predicates is diophantine. The following lemmata are just here for the purpose to close exactly this gap.

Lemma 6 *The functions $(x,y) \mapsto x^y$, $(x,y) \mapsto \binom{x}{y}$, and $x \mapsto x!$, are diophantine.*

Proof: We omit the proofs because they are technical. They can be found in [15] or in [20].

Lemma 7 *Let $P \in \mathbb{Z}[b,z,x]$. Let n be the total degree, and let m be a bound for the sum of the coefficients. Then the following formulas are equivalent over $\mathbb{N}$:*

1. $\forall z_{<u} \exists x P(b,z,x) = 0$;
2. $\exists(y,v,w)(\binom{v}{u} \mid P(b,v,w) \wedge$
   $\frac{v-u}{u-1} > m(b+u+y+1)^{ny} \wedge (u!)^2 \mid v+1 \wedge \binom{v}{u} \mid \frac{w!}{(w-y)!})$.

Proof: Assume the first formula is true. Let $x_0, \ldots, x_{u-1}$ be such that $P(b,z,x_z) = 0$ for $z = 0, \ldots, u-1$. Let y be such that $x_z < y$ for all x_z. Let v be such that the second and third condition are fulfilled, i.e. v is congruent -1 modulo $(u!)^2$ and $v > u + (u+1)m(b+u+x+1)^{ny}$.

For $z = 0, \ldots, u-1$, the numbers $u_z := \frac{v+1}{z+1} - 1 = \frac{v-z}{z+1}$ are integers, and they are pairwise relatively prime: if d is a common prime factor of u_i and u_j for $i < j < u$, then d divides $(j+1)u_i - (i+1)v_j = j - i$, hence $d < u$, hence $d \mid u!$, hence $d \mid \frac{v+1}{i+1}$; because also $d \mid u_i = \frac{v+1}{i+1} - 1$, we get $d = 1$.

By the Chinese remainder theorem, there exists w such that w is congruent x_z modulo u_z, for $z = 0, \ldots, u-1$. Moreover, we have that v is congruent z modulo u_z. Because $P(b,z,x_z) = 0$, it follows that $P(b,v,w)$ is congruent 0 modulo u_z. Since the u_z are pairwise relatively prime, $P(b,v,w)$ is divisible by their product, which is equal to $\binom{v}{u}$. This shows that the first condition is fulfilled.

For $z = 0, \ldots, u-1$, the number $w - x_z$ is divisible by u_z. Since $x_z < y$, also $\frac{w!}{(w-y)!} = w(w-1)\ldots(w-y+1)$ is divisible by u_z. Again because the u_z are pairwise relatively prime, $\frac{w!}{(w-y)!}$ is divisible by their product $\binom{v}{u}$. This shows that the fourth condition is fulfilled.

Now, assume that the second formula is true, i.e. we have y, v, w such that the four conditions hold. Let z be an arbitrary number less than u. Again, we define $u_z := \frac{v-z}{z+1}$. From the fourth condition, we conclude that u_z divides the product $w(w-1)\dots(w-y+1)$. Therefore we can factor u_z into a product of y numbers, each dividing one factor $w-j$.

There must be a factor r of u_z and a factor $w-x$ of the right product such that $r^y \geq u_z$. Since $v-z$ is divisible by u_z and therefore divisible by r, we get v congruent z modulo r. Since $w-x$ is divisible by r, w is congruent x modulo r. By the first condition, $P(b, v, w)$ is congruent 0 modulo u_z, and therefore congruent 0 modulo r. Hence $P(b, z, x)$ is congruent 0 modulo r.

To turn the modulo equation into an absolute equation, we use a bound for the absolute value of the polynomial. In fact,

$$|P(b, z, x)| < m(b+z+x+1)^n < m(b+u+y+1)^n < u_z^{1/y} \leq r,$$

where the third inequality is a consequence of the second condition. This implies $P(b, z, x) = 0$.

Because z was chosen arbitrary, we can construct such an x for each z less than u, and the first formula is true.

Lemma 8 *Bounded universal quantification of diophantine predicates is diophantine.*

Proof: For the case of one free variable and one existential quantifier, lemma 7 shows the claim (the second predicate is diophantine by the previous formula). The general case follows from a generalization of lemma 7 which is proven similarily.

Theorem 38 (Matijasevic) *Any recursively enumerable set is diophantine.*

Proof: Replace “arithmetic” by “diophantine” in the proof of theorem 35 and corollary 6.

If P is a unary recursively enumerable set which is expressed by

$$P(x) :\leftrightarrow \exists(y_1, \dots, y_m)(Q(x, y_1, \dots, y_m) = 0),$$

then the $m+1$-ary polynomial function defined by the polynomial $x(1-2Q^2)$ has value x if $x \in P$ and is undefined (since the value is negative) otherwise. Therefore, we have as a consequence:

Corollary 8 *Any recursively enumerable subset of $\mathbb{N}$ is the image set of a polynomial partial function.*

Remark 10 For a long time, experts in number theory believed that there is no "formula for primes", i.e. no expression with all values being prime numbers. By the above corollory, there exists even a polynomial, and the value set is the set of all prime numbers.

Assume that we have a coding function of the set of all polynomials with integer coefficients to $\mathbb{N}$, subject to the conditions

- the coding is un-ambiguous (i.e. the function is injective);
- the code of a polynomial can be recursively computed from the coefficients.

It is easy to construct such codes.

Theorem 39 (recursive unsolvability of Hilbert's 10th problem) *The unary predicate "x is the code of a polynomial which has a zero in the natural numbers" is not recursive.*

Proof: Let P be a unary predicate which is recursively enumerable but not recursive (it was shown in a previous section that such sets exist). By Matijasevic's theorem, there is a polynomial Q such that

$$P(x) \leftrightarrow \exists(y_1, \ldots, y_m)(Q(x, y_1, \ldots, y_m) = 0).$$

Suppose indirectly that there exists a recursive program X that decides whether the input is the code of a polynomial which has a zero. Then the following program with input u,

> compute the code p_u of the polynomial $Q[x := u]$ in $y_1, \ldots, y_m$;
> apply X to p_u

would decide recursively whether $u \in P$, a contradiction.

Remark 11 The original problem asked for an algorithmic method for deciding the existence of a solution in integers, not in natural numbers. But also this problem is recursively unsolvable, because $Q(x_1, \ldots, x_n)$ has a solution in integers iff the polynomial

$$\prod_{i_1,\ldots,i_n=0,1} Q((-1)^{i_1}x_1, \ldots, (-1)^{i_n}x_n)$$

has a solution in the natural numbers.

11 λ-calculus

In λ-calculus, one tries to formalize functions. Therefore, there is no metatheoretic distinction between informal functions and objects, because the objects are the informal functions. Terms can be built using only one formal function symbol: application. Instead of $f(x)$, we write $app(f, x)$ (where a is application).

Functions with several arguments are handled in the following way. There is a one-to-one correspondence between functions binary functions unary functions with values in the unary function space: for any binary function f and value x, let f_x be the unary function obtained by replacing the first argument by x. This assignment $x \mapsto f_x$ gives one direction. Reversely, for any function valued function g, one can construct a binary function by $(x, y) \mapsto app(app(f, x), y)$. By this isomorphism, we may consider any function as a univariate function.

We make the convention to write $app(f, x)$ as fx, binding from the left, i.e. $xyz = (xy)z = app(app(x, y), z)$. This has the advantage that terms built by application look like usual function terms in polish notation, e.g. the term corresponding infomally to $a+b\cdot c$, which is $app(app(+, a), app(app(\cdot, b), c))$, can be written $+a(\cdot bc)$.

λ-calculus is an equational theory with quantifiers. For any term t and any variable x, there is a quantified term λxt or $\lambda x.t$, meaning the function mapping x to t. We make the convention that λ binds weaker than application, i.e. $\lambda x.yz = \lambda x.(yz)$. We write $\lambda x_1 \ldots x_n.t$ as a shorthand for $\lambda x_1.\lambda x_2.\ldots.\lambda x_n.t$. The free variables of a term are the variables which are not bounded by a λ-quantifier. A term is closed iff it does not have free variables. Closed terms are also called combinators and often denoted by capitalized letters. For any term, its closure is obtained by quantifying all free variables.

The axioms are the following:

equality axioms , except that the substitution rule must be restricted to the case where the variable is free for substitution of the term;

α-rule or renaming of bound variables: $\lambda x.t = \lambda y.t[x := y]$, if x is free for y in t;

β-rule : $(\lambda x.p)q = p[x := q]$, if x is free for q in p.

Sometimes, one also wants to adjoin the following axiom:

η-rule : $\lambda x.tx = t$, if x is not free in t.

This axiom is often called extensionality (a function can be recovered from its action on the arguments). In this section, we consider λ-calculus without η-rule.

Here are some prominent combinators. The identity combinator is defined as $I := \lambda x.x$. The "loop forever" combinator $\Omega := (\lambda x.xx)\lambda x.xx$ has the property that it reduces to itself by a β-conversion. The combinators $K := \lambda xy.x$ and $S := \lambda xyz.xz(yz)$ are famous because every combinator can be expressed by K,

S, and application. The most famous combinator is introduced in the following theorem.

Theorem 40 (Fixed point theorem) *There is a combinator Y, such that for every f, we have $f(Yf) = Yf$, i.e. Yf is a fixed point of f.*

Proof: Set $Y := \lambda y.(\lambda x.y(xx))\lambda x.y(xx)$. Then we have $Yf = (\lambda x.f(xx))\lambda x.f(xx)$, which is equal to $f((\lambda x.f(xx))\lambda x.f(xx)) = f(Yf)$ by the β-rule.

In order to get normal forms, we orient the β-rule from left to right. The α-rule cannot be oriented, but α-convertible terms differ only by a renaming of variables, so we identify α-convertible terms on a syntactic level.

We write $t_1 \xrightarrow{\beta} t_2$ iff there is a subterm s_1 of t_1 which reduces by the β-rule to a subterm s_2 of t_2, and the context s_1 in t_1 is equal to the context of s_2 in t_2. Obviously, provable equality in the λ-calculus is the reflexive symmetric transitive closure of $\xrightarrow{\beta}$.

Note that we have

$$Yf \xrightarrow{\beta} f(Yf) \xrightarrow{\beta} f(f(Yf)) \xrightarrow{\beta} f(f(f(Yf))) \xrightarrow{\beta} \ldots.$$

This example also shows that β-reduction is not noetherian.

Unfortunately, $\xrightarrow{\beta}$ does not have the diamond property, as shown by the following example.

Let $t_1 := (\lambda x.xx)\lambda u.v$. By reducing the subterm $\lambda u.v$ we obtain $t_2 := (\lambda x.xx)v$, and by reducing the first lambda, we get $t_3 := (\lambda u.v)\lambda u.v$. By reducing t_2, we get the normal form $t_4 := vv$. However, we do not have $t_3 \xrightarrow{\beta} t_4$, because we need two reduction steps to reduce t_3 to t_4.

Here are some properties of the β reduction.

Theorem 41 *The reduction $\xrightarrow{\beta}$ is substitutive.*

Proof: Let $t_1 = (\lambda x.p)q \xrightarrow{\beta} t_2 = p[x := q]$ be a β-reduction. Let y be a variable and r a term such that y is free for r in t_1, t_2. We distinguish two cases:

Case 1: y is different from x. Then we have $t_1[y := r] = (\lambda x.p[y := r])q[y := r]$. On the other hand, we have $t_2[y := r] = p[x := q][y := r] = p[y := r][x := q[y := r]]$, hence $t_1[y := r] \xrightarrow{\beta} t_2[y := r]$.

Case 2: $y = x$. Then $t_1[x := r] = (\lambda x.p)q[x := r]$, because x is not free in $\lambda x.p$. On the other hand, $t_2[x := r] = p[x := q][x := r] = p[x := q[x := r]]$, hence we have again $t_1[x := r] \xrightarrow{\beta} t_2[x := r]$.

Theorem 42 *The reduction $\xrightarrow{\beta}$ is weakly Church-Rosser.*

Proof: Suppose that t_0 has two reducible subterms (redexes) $s_1 = (\lambda x.p_1)q_1$ and $s_2 = (\lambda y.p_2)q_2$. Let t_1 be the result obtained by reducing s_1 to $s_1' := p_1[x := q_1]$, and let t_2 be the result obtained by reducing s_2 to $s_2' := p_2[x := q_2]$. We have to show that t_1 and t_2 have a common reductum. The following cases have to be considered.

Case 1: s_1 and s_2 are disjoint. Then the reduction do not effect each other, and it is easy to close the diamond.

Case 2: s_1 is a subterm of p_2, i.e.

$$t_0 = \ldots((\lambda y.\ldots s_1\ldots)q_2)\ldots,$$

$$t_1 = \ldots((\lambda y.\ldots s_1'\ldots)q_2)\ldots,$$

$$t_2 = \ldots(\ldots s_1\ldots)[y := q_2]\ldots.$$

Then t_1 reduces to

$$t_3 = \ldots(\ldots s_1'\ldots)[y := q_2]\ldots.$$

By substitutivity, t_2 also reduces to t_3.

Case 3: s_2 is a subterm of q_1, i.e.

$$t_0 = \ldots((\lambda y.p_2)(\ldots s_1\ldots))\ldots,$$

$$t_1 = \ldots((\lambda y.p_2)(\ldots s_1'\ldots))\ldots,$$

$$t_2 = \ldots p_2[y := (\ldots s_1\ldots)]\ldots.$$

Then t_1 reduces to

$$t_3 = \ldots p_2[y := (\ldots s_1'\ldots)]\ldots.$$

By induction on the structure of a term p, one proves easily that $q \xrightarrow{\beta} q'$ implies $p[x := q] \xrightarrow{\beta*} p[x := q']$ (in general, several reduction steps are required, according to the number of free occurences of x). It follows that t_2 also reduces to t_3.

The remaining cases are analogous.

The following theorem is more difficult. The book [1] discusses several proofs.

Theorem 43 *The reduction $\xrightarrow{\beta}$ is Church-Rosser.*

Corollary 9 *The λ-calculus is consistent.*

Proof: It suffices to exhibit two distinct normal forms. These are I and (for instance) $\lambda x(xx)$.

Not every λ-term has a normal form (think of Ω). But if t has a normal form, then there exists a reduction to it by corollary 1. Here is a theorem that tells how to find such a normal form.

Theorem 44 *If a λ-term has a normal form, then one obtains a reduction to the normal form by reducing always the leftmost λ for which a reduction is possible.*

Proof: See [1].

Example 4 The λ-term $(\lambda y.z)\Omega = (\lambda y.z)((\lambda x.xx)(\lambda x.xx))$ reduces to z by reducing the left λ-term. The right λ-term Ω may also be reduced, but this reduction leads to an infinite loop.

The proof of the following result is also beyond the scope of this lecture.

Theorem 45 (Böhm) *If t_1 and t_2 are two distinct normal forms, then the new axiom $t_1 = t_2$ leads to an inconsistency.*

By Böhm's theorem, one has to be careful when adding new axioms. In order to formalize informal theories in a reasonable way, it is necessary to introduce new constants. Doing this, we get *applied λ-calculus.*

Example 5 In some sense, we already gave *lambda*-theories when we introduced a name for a combinator. Namely, we added a new constant (e.g. I) and a new axiom ($I = \lambda x.x$).

Example 6 A more interesting example is the following. We add the constants *mult*, *one*, *inv*, and the axioms

1. $mult(mult\ xy)z = mult\ x(mult\ yz)$,
2. $mult\ x\ one = mult\ one\ x = x$,
3. $mult\ x(inv\ x) = mult\ (inv\ x)x = one$.

In many applications, the constants - denoting functions - come with a natural type information. This motivates to consider a variant of λ-calculus, called typed λ-calculus.

Here we have a collection of types with the following property: if A and B are types, then we also have a type $A \to B$ (the type of functions from A to B). Every term has a unique type. We write $t : A$ for the statement t has type A.

Term forming by application is allowed only if the types match; more precisely, st is defined only if $s : A \to B$ and $t : A$, and we have $st : B$. λ-abstraction is unrestricted; if $x : A$ and $t : B$, then $\lambda x.t : A \to B$. The axioms for λ-calculus hold whenever both sides are defined (it follows that both sides have the same type).

Example 7 The new axioms added in example 6 can be typed: one sets $one : G$, $inv : G \to G$, $mult : G \to G \to G$.

The reduction $\overset{\eta}{\to}$ is Noetherian and Church-Rosser in the typed case. Therefore, we have decidable equality. Unfortunately, we may loose decidability by adding new axioms.

Some λ-terms are *polymorphic*, i.e. they can have several types. In this case, there is a most general type with the propery that any possible type as an instantiation. For instance, the identity function $\lambda x.x$ has most general type $A \to A$, where A can be an arbitrary type. There is an algorithm that computes a most general type, if exists, for a given λ-term (type inference).

Example 8

- The most general type of $K = \lambda xy.x$ is $A \to B \to A$.
- The most general type of $S = \lambda xyz.xz(yz)$ is
$$(A \to B \to C) \to (A \to B) \to A \to C.$$
- The λ-term xx cannot by typed.

The λ-calculus can be used as a programming language. A program consists of a sequence of definitions of the form NAME = λ-TERM and of a single 'command', which is just a λ-term. The interpreter executes the command, using the given definitions. The definitions could in principle be eliminated, but they make programs shorter and better readable.

'Executing a command', i.e. a λ-term, means to reduce it by β-reduction (or some other reduction coming from an axiom) until a normal form is obtained. As we saw earlier, the order of reduction is important and may effect termination. A reduction order which always reduces an innermost reducible subterm is called *applicative order*. This is essentially the usual method used in procedure and function calls in most imperative and most programming languages: first we evaluate the arguments of a procudure or a function, and then we call the procedure/function. This method is called *call by value*.

Another reduction order is *normal order*, which always reduces an outermost reducible subterm. This order is used in Haskell (in case of several outermost subterms, the leftmost is chosen); it is also similar to the method *call by name* that has been used in the imperative language Algol60. As we saw, it always leads to a normal form if there exists one.

Normal order reduction allows to manipulate 'infinite objects', i.e. λ-terms without normal form, within programs.

Example 9 We use a typed λ-calculus with certain pre-defined types, constants and axioms (i.e. an applied λ-calculus).

```
type Bool
const True: Bool
const False: Bool
const Not: Bool -> Bool
axiom Not True = False
axiom Not False = True
const IfThenElse: Bool -> A -> A -> A
axiom IfThenElse True x y = x
axiom IfThenElse False x y = y

const FixedPoint: (A -> A) -> A

type List(A)
const Null: List(A)
const Head: List(A) -> A
const Tail: List(A) -> List(A)
const Put: A -> List(A) -> List(A)
axiom Head Put x y = x
axiom Tail Put x y = y

type Nat
const 0,1,2,3,...: Nat
const +,*,-: Nat -> Nat -> Nat
const IsDivisibleBy: Nat -> Nat -> Bool
const ==: Nat -> Nat -> Bool
const length: List(A) -> Nat
```

The constants $+$, $*$, `==`, and `IsDivisibleBy` will be written in infix notation to improve readibility. They are supposed to fulfil the usual axioms. We also write `(a,l)` instead of tt Put a l.

The `FixedPoint` function allows to do recursive definitions. For instance, we can express the recursion

```
fac: Nat -> Nat
fac = \n IfThenElse n == 0 0 n*fac(n-1)
```

by

```
fac = FixedPoint \f \n IfThenElse n == 0 0 n*f(n-1)
```

From now on, we will write the `IfThenElse` by the usual 'if .. then .. else' construct, and we will use recursive definitions as usual.

Here are some auxiliary definitions. `take i l` takes the first i elements from the list l. `from n` produces the infinite list $n, n+1, n+2, \ldots$. `filter p l` constructs the sublist of the elements fulfilling property p.

```
take: Nat -> List(A) -> List(A)
take = \n \l if n == 0 then Null else ( Head l, take (n-1) Tail l)

from: Nat -> List(Nat)
from = \n ( n, from (n+1))

filter: (a -> Bool) -> List(a) -> List(a)
filter = \p \l if Length(l) == 0 then Null else
if p Head l then ( Head l, filter p Tail l) else filter p Tail l
```

Note that the command `from 2` causes an infinite loop, but the command `take 4 from 2` gives the answer $[2, 3, 4, 5]$.

In order to find the first 10 primes, we use the sieve of Erastothenes.

```
sieve: List(Nat) -> List(Nat)
sieve = \l ( Head l, sieve filter Not DivisibleBy Head l Tail l)

primes: List(Nat)
primes = sieve from 2

take 10 primes
```

The answer is $2, 3, 5, 7, 11, 13, 17, 19, 23, 29$.

12 Higher Order Logic

In this section, we describe a particular applied λ-calculus that models higher order logic. It should be emphasized that this is only one possible formalization, which is designed in order to emphasize the analogy and the differences to the given system of first order logic.

We have the type *Bool* and constants $\top, \bot : Bool$, $\neg : Bool \to Bool$, $\wedge, \vee, \to, \leftrightarrow : Bool \to Bool \to Bool$. As usual, we write the binary functions infix. The binding conventions are the same as in the previous sections.

The existential and universal quantifiers are polymorphic constants $\forall, \exists : (A \to Bool) \to Bool$. If A term of type boolean, and x is a variable of type A, then we cannot simply form the expression $\forall x A(x)$; rather, we first form a predicate from A by λ-abstraction and then apply it to the $\forall$ function: $\forall \lambda x.A$. After this has been clarified, we introduce the usual notation $\forall x A(x)$ as a syntactic sugar for $\forall \lambda x.A$.

In order to deal with 'local' equality, we introduce polymorphic constants "=" of type $A \to A \to Bool$ and "ifthenelse" of type $Bool \to A \to A \to A$.

Here are the axioms.

1. $A \to B \to A = \top$
2. $(A \to B \to C) \to (A \to B) \to (A \to C) = \top$
3. $(\neg A \to \neg B) \to (B \to A) = \top$
4. $\forall \lambda x.A \to A[x := t] = \top$, if x is free for t in $A(x)$
5. $\forall x(C \to A(x)) \to C \to \forall x A(x) = \top$, if x is not free in C
6. $\vee = \lambda AB.(\neg A \to B)$
7. $\wedge = \lambda AB.\neg(A \to \neg B)$
8. $\leftrightarrow = \lambda AB.((A \to B) \wedge (B \to A))$
9. $\bot = \neg\top$
10. $\exists = \lambda P.\neg\forall\lambda x.\neg(Px)$
11. $\top \to A = A$
12. $\forall \lambda x.\top = \top$
13. $(A \to B) \to B = (B \to A) \to A$
14. $(x = y) = (y = x)$
15. if T then x else $y = x$
16. if F then x else $y = y$

17. if $(x = y)$ then x else $y = y$

Moreover, we adjoint the η-rule to the list of axioms.

Because HOL is an equational theory, we start by proving the replacement theorem.

Theorem 46 *Let $A, B : Bool$ such that $A \to B = \top$ and $B \to A = \top$. Then $A = B$.*

Proof:

$$A = \top \to A = (B \to A) \to A = (A \to B) \to B = \top \to B = B.$$

Similarily, one can show that if $(x = y) = T$ then $x = y$ (using the axioms for "if then else").

It is obvious that any formula in first order predicate logic can be translated into our system. Any such formula is a term of type boolean. The following states that proofs get preserved by this translation.

Theorem 47 *Let Γ be a set of first order formulae, and let A be a first formula in first order formula such that $\Gamma \vdash_{FOL} A$. Then the formula $A = \top$ can be derived by higher order logic from the set of formulae $\{B = \top \mid B \in \Gamma\}$.*

Proof: We may replace every formula in $\Gamma \cup \{A\}$ by an equivalent formula containing only $\neg, \to, \exists$; the resulting formulae are λ-convertible, so a derivation of the replaced formula by the replaced assumption is possible if and only if a derivation of the original formula is possible. Therefore we may restrict our attention to the kernel language.

For every first order axiom A in the kernel language, we have a higher order axiom $A = \top$.

Suppose that we have derived A by modus ponens from $B \to A$ and B. By induction hypothesis, there is a HOL derivation of $B \to A = \top$ and of $B = \top$. Therefore we obtain

$$A = (\top \to A) = (B \to A) = \top.$$

Suppose that we have derived $A = \forall \lambda x.B$ by generalization from B. By induction hypothesis, there is a HOL derivation of $B = \top$. Then we get

$$\forall \lambda x B = \forall \lambda x \top = \top.$$

An interpretation for typed λ calculus is given by a set $S(A)$ for each type A, such that $S(A \to B) = S(B)^{S(A)}$, the set of all functions from $S(A)$ to $S(B)$; and an element in $S(A)$ for each constant and variable of type A. These data induce a valuation of λ-terms, such that the value of a term of type A is an element of $S(A)$. For compound terms, the valuation is defined recursively:

- let $p : A \to B$, $q : A$, then $v(pq) := v(p)v(q)$ (the function $v(p)$ applied to the argument $v(q)$);
- let $x : A$, $t : A$, then $v(\lambda x.t)$ is the function from $S(A)$ to $S(B)$ that maps $a \in A$ to $v_{x/a}(t)$, where $v_{x/a}$ is the valuation that coincides with v in all variables and constnats except for the value of x, which is a.

In the case of HOL, we set $S(B) = \{T, F\}$, and the values for $\neg, \bot, \wedge, \vee, \to, \leftrightarrow$ are defined as usual. For every type A, the value of $\forall_A$ is the function that maps a predicate $p \in \{0,1\}^{S(A)}$ to T iff the predicate is always true. Likewise, the value of $\exists_A$ is the function that maps a predicate $p \in \{0,1\}^{S(A)}$ to F iff the predicate is always false. The value of the constant " $=''$ is the diagonal, and the value of the constant "if then else" is the function that maps (T, x, y) to x and (F, x, y) to y.

Lemma 9 *The value of a λ-term depends only on the values of its free variables.*

Theorem 48 *If a formula can be proven by HOL, then it is holds in any interpretation (i.e. the value of the left hand side is equal to the value of the right hand side).*

Proof: The proof is tedious but not difficult. It suffices to show that every axiom holds in any interpretation.

The following corollary states that HOL subsumes FOL.

Theorem 49 *Let Γ be a set of first order formulae, and let A be a first formula in first order formula. Then $\{B = \top \mid B \in \Gamma\} \vdash_{HOL} A = \top$ iff $\Gamma \vdash_{FOL} A$.*

Proof: If: this has already been proven above.

Only if: Suppose that $\{B = \top \mid B \in \Gamma\} \vdash_{HOL} A = \top$. Let $\mathcal{I}$ be a FOL interpretation of Γ. Then we can extend $\mathcal{I}$ to an HOL interpretation $\mathcal{I}'$ of $\{B = \top \mid B \in \Gamma\}$. The value of the formulas in $\Gamma \cup A$ do not depend on the extension, because all constants and variables occuring in Γ are of type O or of type $O \to O$ or of type $O \to Bool$, and for those three types all values are defined in $\mathcal{I}$. By the correctness of HOL, any HOL interpretation that satisfies $\{B = \top \mid B \in \Gamma\}$ also satisfies $A = \top$. Thus, $\mathcal{I}' \models A = \top$. By restricting valuation to FOL terms again, it follows that $\mathcal{I} \models A$. We have shown that any FOL interpretation of Γ also satisfies A.

By the completeness theorem of FOL, it follows that $\Gamma \vdash_{FOL} A$.

By lemma 9, the value of a formula without any free variables depends only on the sets $S(A)$. The sets for function types are completely determined by the sets of their argument types, so the values actually depend only on the sets $S(A)$ of non-function types A. If we have only two non-function types *Bool* and O (the type of 'objects'), then the value of a non-closed formula determines only on the cardinality of $S(O)$, because two sets having the same cardinality are isomorphic.

Example 10 Below we construct a formula without free variables that has value T if $S(O)$ is countable, and F otherwise.

$$x, y : O, f : O \to O, A, B : O \to Bool$$

$$Func, Inj, Surj, Bij : (O \to O) \to (O \to Bool) \to (O \to Bool) \to Bool$$

$$Func = \lambda f AB.\forall x A(x) \to B(fx)$$

$$Inj = \lambda f AB.\forall x \forall y(A(x) \to A(y) \to (fx = fy) \to (x = y))$$

$$Surj = \lambda f AB.\forall y \exists x(B(y) \to (A(x) \wedge (fx = y)))$$

$$Bij = \lambda f AB.(Inj(f, A, B) \wedge Surj(f, A, B))$$

$$Inf : (O \to Bool) \to Bool$$

$$Inf = \lambda A.\exists f(Func(f, A, A) \wedge Inj(f, A, A) \wedge \neg Surj(f, A, A))$$

$$EqCard : (O \to Bool) \to (O \to Bool) \to Bool$$

$$EqCard = \lambda AB.\exists f(Func(f, A, B) \wedge Bij(f, A, B))$$

$$Universe : O \to Bool$$

$$Universe = \lambda x.T$$

$$Count : Bool$$

$$Count = Inf(Universe) \wedge \forall A(Inf(A) \to EqCard(A, Universe)$$

Clearly, HOL is much more expressive than FOL. The other side of the medal is that we cannot expect to have a complete logical system.

Theorem 50 *The given axiom system for HOL is incomplete: there are formulae which are true in any HOL interpretation but cannot be proven in HOL. Moreover, there is no complete axiomatization.*

Proof: Assume, indirectly, that there is an axiomatization of HOL. Then the set of all formulae that hold in any interpretation is semi-decidable. Then we claim that we can also decide whether a given formula in FOL is a theorem, and this is known to be undecidable.

Here is an algorithm that decides whether a given closed FOL-formula A is a theorem in FOL.

1. Let B be a universal closure of A in the HOL system with non-function types O and *Bool*.
2. Let *Count* be a formula which has value true iff $S(A)$ is countable. Then either the value of $Count \to B$ is true in any HOL model, or the value of $C \to \neg B$ is true in every HOL model, depending on the value of B in the countable model.
3. Determine which of the two alternatives in step 2 holds, by semi-deciding both possibilities in parallel.
4. In the first case, answer "yes". In the second case, answer "no".

The algorithm actually tells us if the formula A is true in every countable interpretation or not. If A is a FOL theorem, then it clearly is true in any countable interpretation. If A is not a FOL theorem, then its negation has a countable model by FOL completeness, and therefore is is not true in every countable interpretation.

13 Constructive Logic

The language of constructive logic is the same as the language of classical logic. The inference rules are also the same, the axioms are almost the same: axiom 10 is replaced by the axiom $A \to \neg A \to B$. The main difference is a semantical one: the logical connectives and the quantifiers are interpreted in a different way.

Informally speaking, a formula is considered true iff it is "witnessed" somehow, e.g. by a proof. We do not make this witnessing more precise at the moment. The logical connectives and quantifiers are then interpreted in the following way.

- A proof of $A \wedge B$ is given by a proof of A and a proof of B.
- A proof of $A \vee B$ is given by a pointer to A or B and a proof of the subformula pointed at.
- A proof of $A \to B$ is given by a construction producing a proof of B when a proof of A is given.
- There is no proof of $\bot$.
- A proof of $\neg A$ is given by a construction producing a proof of $\bot$ when a proof of A is given.
- A proof of $\forall x A(x)$ is a construction producing a proof of $A(c)$ when an object c is given.
- A proof of $\exists x A(x)$ is given by an object c and a proof of $A(c)$.

For instance, a proof of the formula $\forall x \exists y A(x, y)$ consists of a construction producing, for given x, a y and a proof for $A(x, y)$.

The question "what precisely is a construction?" has been answered in different ways. One possible answer is to state "a construction is a recursive function" or equivalently "a construction is something that can be computed by a Turing machine". The intuitionists answer this question negatively: a construction is a free creation of mind, and any precise definition would therefore limit the scope and is therefore already inadequate.

Interestingly enough, a large part of constructive mathematics can be developed without an explicit answer to the above question.

Remark 12 We observe that the $A \to \bot$ and $\neg A$ have the same interpretation. Indeed, $\neg$ can be defined in constructive logic. This is the only example where one logical connective can be defined in others.

In contrast to this, we have much more inter-definability in classical logic. There, any connective can be defined in terms of $\to$ and $\bot$.

On the ground of the interpretation, one can easily give proofs for all axioms.

Example 11 In the following, we write $a \models A$ for "a is a proof of A". The pointers to the first and second part of a disjunctive formula are denoted by p_1, p_2. If a proof a has two parts (e.g. the proof of a conjunction), then $q_1(a), q_2(a)$ are these two parts.

1. For $a \models A$, the construction λba constructs a proof of a for any proof b of B. Hence $\lambda a \lambda ba$ is a proof for $A \to B \to A$.
2. The construction q_1 is a proof for $A \wedge B \to A$.
3. Any construction serves as a proof of $\perp \to A$ (since it is applied to the empty domain).

An important non-example is the principle of the excluded middle $A \vee \neg A$. A proof of this disjunction is a pointer to the either A or $\neg A$ and a proof of this pointed subformula. However, there are a lot of examples of propositions A for which we do not have such a proof. Thus, $A \vee \neg A$ is not universally valid in constructive logic.

On the other hand, one can show constructively that the negation of the principle of the excluded middle is inconsistent: Let $c \vdash \neg(A \vee \neg A)$. If $d \vdash (A \vee \neg A)$, then $c(d) \vdash \perp$. If $e \vdash A$, then $g := \lambda ec(p_1, e)$ is a proof for $A \to \perp = \neg A$. If $f \vdash \neg A$, then $h := \lambda fc(p_2, f)$ is a proof for $\neg A \to \perp$. Hence $h(g)$ proves $\perp$. The construction $c \mapsto h(g)$ as above is a proof for $\neg\neg(A \vee \neg A)$.

Another non-example is the formula $\neg \forall x\ A(x) \to \exists x\ \neg A(x)$. There are example of formulas for which we know constructively that $\forall x\ A(x)$ is inconsistent, but we do not have an x_0 for which $\neg A(x_0)$, and so we cannot prove $\exists x\ \neg A(x_0)$.

In the following, we describe a way for embedding classical predicate logic into constructive predicate logic.

A formula is called negative iff all atoms are negated and it does not contains $\vee$ or $\exists$.

It is not hard to define a "negative translation" $A \mapsto F(A)$ with the following properties:

- $F(A)$ is negative;
- we have $\vdash A \leftrightarrow F(A)$ in classical logic.

To achieve this, we replace any disjunction $A \vee B$ by $\neg(\neg A \wedge \neg B)$, any existential quantifier $\exists x A(x)$ by $\neg \forall x \neg A(x)$, and any atom P by $\neg\neg P$.

Lemma 10 *If A is negative, then* $\vdash A \leftrightarrow \neg\neg A$.

Proof: For any formula A, we have $\vdash A \to \neg\neg A$ (exercise). The other direction is shown by induction on the structure of A. We have to distinguish the following cases:

- A is of the form $\neg B$ for an atom B. Because one can show $\neg\neg\neg B \to \neg B$ (exercise), the statement holds for this case.
- A is of the form $\neg A$ with A negative. This case can be treated as the previous one.
- A is of the form $A \wedge B$ with A, B negative. One can show $\neg\neg(A \wedge B) \to \neg\neg A \wedge \neg\neg B$. Because A and B are negative, we have $\neg\neg A \to A$ and $\neg\neg B \to B$ by induction hypothesis. Thus, we get $\neg\neg(A \wedge B) \to (A \wedge B)$.
- A is of the form $A \to B$ with A, B negative. One can show $\neg\neg(A \to B) \to (A \to \neg\neg B)$. Because B is negative, we have $\neg\neg B \to B$ by induction hypothesis. Thus, we get $\neg\neg(A \to B) \to (A \to B)$.
- A is of the form $\forall x A(x)$ with $A(x)$ negative. One can show $\neg\neg\forall x A(x) \to \forall x \neg\neg A(x)$. By induction hypothesis, $\neg\neg A(x)$ is equivalent to $A(x)$, hence $\neg\neg\forall x A(x) \to \forall x A(x)$.

Theorem 51 *Suppose that $A_1, \ldots, A_n \vdash C$ in classical logic. Then $F(A_1), \ldots, F(A_n) \vdash F(C)$ in constructive logic.*

Proof: It suffices to show that $F(A)$ is a theorem for every classical axiom, and that the above transformation of proofs works for every inference rule.

For the classical axioms 1-5, 9, 11, 13, the negative translation is of the same axiom scheme. Hence this are again axioms and therefore provable.

Axiom 6: The translation is $F(A) \to \neg(\neg F(A) \vee \neg F(B))$, and this is provable in constructive logic.

Axioms 7,12: analogously.

Axiom 10: The translation is $\neg\neg F(A) \to F(A)$, and this is provable because $F(A)$ is negative.

Axiom 8: The translation is

$$(F(A) \to F(C)) \to (F(B) \to F(C)) \to \neg(\neg F(A) \wedge \neg F(B)) \to F(C).$$

One can show

$$F(A) \to F(C), F(B) \to F(C), \neg(\neg F(A) \wedge \neg F(B)), \neg F(C) \vdash \perp.$$

By the deduction theorem (which holds also in constructive logic, by the same proof), we can derive $\neg\neg F(C)$ from the first three hypotheses. But $F(C)$ is negative, hence we can also derive $F(C)$. By the deduction theorem applied three times, we get a proof of the translation.

Now for the inference rules. The translation of the first two inference rules is again of the same pattern, hence they are again valid. For the third inference rule, we have to show

$$F(A(x)) \to F(C) \vdash \neg\forall x \neg F(A(x)) \to F(C),$$

where x does not occur free in $F(C)$. One can derive

$$F(A(x)) \to F(C), \neg\forall x \neg F(A(x)), \neg F(C) \vdash \bot.$$

Thus, $\neg\neg F(C)$ is a logical consequence of the first two hypotheses. Since $F(C)$ is negative, $F(C)$ is a consequence. By the deduction theorem, the validity of the inference rule is proven.

Constructive arithmetic is the theory with the same signature and axioms as formal number theory (the only difference is the logical axiom 10). It is easy to show that for any axiom A, the negative translation is provable.

Lemma 11 *The following formulas are provable:*

$$x = y \vee \neg x = y, \neg\neg x = y \to x = y.$$

Proof: Let $Q(x, y)$ be the formula $x = y \vee \neg x = y$. Then one can show the theorems $Q(0, 0)$, $Q(0, x) \to Q(0, x')$, $Q(x, 0) \to Q(x', 0)$, and $Q(x, y) \to Q(x', y')$. Then one can show $Q(x, y)$ by a double induction (see [30], proposition 2.6, p 124). The second formula follows from the first by propositional constructive logic.

Theorem 52 *Let A be a formula in constructive arithmetic containing no existential quentifiers and no disjunctions. Then A is provable iff A is provable in classical arithmetic.*

Proof: The "only if" implication is obvious. For the "if" implication, suppose that A is a formula without existential quentifiers and disjunctions that is provable in classical arithmetic. By theorem 51 and by the fact that all axioms translate into theorems, $F(A)$ is provable. By the syntactic structure of A, the only difference in A and $F(A)$ is that the atoms are doubly negated in $F(A)$. But by lemma 11, we have $P \leftrightarrow \neg\neg P$ for any atom P. Hence A is equivalent to $F(A)$, and therefore A is also provable.

Of course, we do not expect constructive arithmetic to be complete, since even classical arithmetic is not. Here is a famous formula scheme—the Markov principle—which is trivially classically provable but not constructively, at least not by the Peano axioms:

$$\forall x(A(x) \vee \neg A(x)) \to \neg\forall x \neg A(x) \to \exists x A(x).$$

Nevertheless, the Markov principle is commonly accepted as a new axiom scheme, on the ground of the following informal proof:

If $a \models \forall x(A(x) \vee \neg A(x))$, then $a(x)$ is a pointer to either $A(x)$ or $\neg A(x)$ together with a proof of the pointed subformula. If $q_1(a(x)) = p_1$, then $q_2(a(x))$ is a proof for $A(x)$. Thus, we can find a proof for $\exists x A(x)$ by the following construction

for x from 0 to ∞ do
if $q_1(a(x)) = p_1$, then return $(x, q_2(a(x)))$; exit loop

from the proof a. This construction terminates, because otherwise $\lambda x\ q_2(a(x))$ would be a proof for $\forall x \neg A(x)$ contradicting the second hypothesis.

Recall our embedding of predicate calculus in λ-calculus in the previous chapter. We can do the same thing in constructive logic, except that the typ of propositions should not be called "boolean", because its signature $\wedge, \vee, \rightarrow, \neg$ does not satisfy the axioms of a boolean algebra. The usual name for an algebra with this signature satisfying the propositional axioms of constructive logic is "Heyting algebra".

Example 12 Let X be a topological space. Let H be the set of all open spaces in X. We define

- $A \wedge B := A \cap B$,
- $A \vee B := A \cup B$,
- $A \rightarrow B := \bigcup\{C \mid C \cap A \subset B\}$,
- $\perp := \emptyset$,
- $\neg A := A \rightarrow \perp$.

Then H with these operations is a Heyting algebra.

The above example gives rise to an un-intended model for constructive logic: take an arbitrary topological space X, and give a typed λ-model with the proposition type being assigned the Heyting algebra above. This model is often useful to refute formula as constructively not provable.

14 Constructive Analysis

As a typical branch of mathematics, analysis is organized in a typed way. Therefore we give a formalization in typed λ-calculus. First, we introduce some conventions for making the language look more familiar.

We usually write $\vdash A$ as a shorthand for $\vdash A = T$, and $\forall_{x:A}P(x)$ as a shorthand for $\forall_X \lambda x P(x)$, and likewise for $\exists$. Moreover, we write the binary logical connectives in infix notation with the usual binding conventions, so that our formulas look very similar to predicate formulas.

Recall that we can define a new function whenever we can prove unique existence. We extend this concept in two ways. First, we can define a partial function f whenever we can prove $A(x,y) \to A(x,z) \to y = z$. We write $B(f(x))$ as a shorthand for $\exists_y A(x,y) \wedge B(y)$, and $E(f(x))$ for $\exists_y A(x,y)$ (the existence predicate). For terms containing partial functions, the conditions and inference rules have to be adapted. For instance, we have to replace axiom 12 by $A(t) \to E(t) \to \exists_x A(x)$, if t contains partial functions.

Second, we can define an operation f whenever we can prove $\exists y\ A(x,y)$. Syntactically, we use f as a function. In the constructive interpretation, $f(x)$ selects just the y that must be provided by the existence proof of $\exists y\ A(x,y)$. In the classical interpretation, f is a Skolem function. In contrast to true functions, operations do not satisfy in general the equality axiom $x = y \to f(x) = f(y)$.

Apart from $\mathbb{N}$, we need the basic type $\mathbb{Q}_+$ of positive rational numbers. This type occurs so often that we reserve the lowercase greek letters for the variables of this type.

If A is a type, then the predicate $D : \mathbb{Q}_+ \to A \to A \to H$ is called a metric for A iff the following axioms are fulfilled:

1. $D\epsilon xy \to D\epsilon yx$,
2. $\delta < \epsilon \to (D\epsilon xy \vee \neg D\delta xy)$,
3. $D\alpha xy \to D\beta yz \to D(\alpha + \beta)xz$,
4. $\forall_\epsilon D\epsilon xy \leftrightarrow x = y$,
5. $\forall_\epsilon(\delta < \epsilon \to D\epsilon xy) \to D\delta xy$,
6. $\exists_\epsilon D\epsilon xy$,

where $x, y, z : A$. Informally, $D\epsilon xy$ corresponds to $d(x,y) < \epsilon$ for a distance function d. We do not use distance functions because the target would be the reals, and up to now it is not yet clear what $\mathbb{R}$ is from a constructive point of view.

We define a binary infix relation $\neq : A \to A \to H$ by

$$x \neq y := \exists_\epsilon \neg D\epsilon xy.$$

Theorem 53 *For the predicate $\neq$, we can show the following theorems:*

1. $x \neq y \to y \neq x$,
2. $x \neq z \to (x \neq y \vee y \neq z)$,
3. $\neg(x \neq y) \leftrightarrow x = y$.

Proof: (1) is trivial.

To show (2), assume we have a proof for $\exists_\epsilon \neg D\epsilon xz$. Then we have ϵ and a proof for $\neg D\epsilon xz$. By metric axiom 2, we have $D\frac{\epsilon}{2}xy$ or $\neg D\frac{\epsilon}{4}xy$. In the second case, the conclusion is shown. Likewise, we have $D\frac{\epsilon}{2}yz$ or $\neg D\frac{\epsilon}{4}yz$. In the second case, the conclusion is also shown. The only remaining case is $D\frac{\epsilon}{2}xy$ and $D\frac{\epsilon}{2}yz$, which implies $D\epsilon xz$ by metric axiom 3. But in this case, we get a contradiction, by the proof of $\neg D\epsilon xz$.

To show (3), observe that the left hand side is logically equivalent to $\forall_\epsilon \neg\neg D\epsilon xy$. This is implied by the right hand side by metric axiom 4. To show the other direction, assume ϵ to be given. By metric axiom 2, we have $D\epsilon xy$ or $\neg D\frac{\epsilon}{2}xy$. Because of the assumption $\forall_\epsilon \neg\neg D\epsilon xy$, the second alternative does not hold, hence we have proven $D\epsilon xy$.

Remark 13 A binary relation fulfilling the axioms above is also called an inequality relation or an apartness relation. Such relations need not exist for arbitrary type. A necessary and sufficient condition for the existence of an apartness relation is the "stability of equality" $\neg\neg x = y \to x = y$. In this case, $\neg(x = y)$ is an apartness relation but usually not the only one. However, one can show that any metric apartness is equivalent to $\neg(x = y)$ under the assumption of Markov's principle.

A fundamental object is a function $r : \mathbb{Q}_+ \to A$ satisfying the condition

$$F(r) := \forall_\epsilon \forall_\delta (\delta < \epsilon \to D(\epsilon - \delta)(r\epsilon)(r\delta)).$$

Informally speaking, r specifies an ϵ-ball with midpoint $r\epsilon$, for each ϵ. The condition $F(r)$ states that these balls are contained one in another.

Trivial examples of fundamental objects are the constant functions $\lambda\epsilon x$. The function $I := K_{A,\mathbb{Q}_+}$ assigns any element in A its constant fundamental object.

We define the binary infix relation $\sim$: $(\mathbb{Q}_+ \to A) \to (\mathbb{Q}_+ \to A) \to H$ as a partial relation restricted to fundamental objects by

$$r \sim s := \forall_\epsilon D(3\epsilon)(r\epsilon)(s\epsilon).$$

Theorem 54 *The relation $\sim$ is an equivalence relation.*

Proof: Reflexivity and symmetry are obvious. To show transitivity, assume $r \sim s$ and $s \sim t$. By $F(r)$ and $F(s)$, we have $D\frac{3\epsilon}{4}(r\epsilon)(r\frac{\epsilon}{4})$, and $D\frac{3\epsilon}{4}(r\epsilon)(r\frac{\epsilon}{4})$. By assumption, we have $D\frac{3\epsilon}{4}(r\frac{\epsilon}{4})(s\frac{\epsilon}{4})$ and $D\frac{3\epsilon}{4}(s\frac{\epsilon}{4})(t\frac{\epsilon}{4})$ By metric axiom 4, it follows $D(3\epsilon)(r\epsilon)(t\epsilon)$.

A metric space (i.e. a type together with a metric) is called complete iff every fundamental object is equivalent to a constant fundamental object.

At this point, we would like to introduce a new type $\overline{A}$, the completion of A with respect to D, whose objects are the equivalence classes of fundamental objects. To define a new type N, we need to do the four things:

1. specify the terms of N,
2. define an equality for N,
3. specify the functions from N to any other type,
4. specify what the quantifiers λ, $\forall$, $\exists$ should do.

Note that it is not necessary to specify the functions into the new type, because any function can be written as a λ-abstraction from a term.

In the case of $\overline{A}$, we proceed as follows.

1. The terms are the terms of $\mathbb{Q}_+ \to A$. Each term becomes partial, and the existential predicate is R.
2. The equality is $\sim$.
3. Any function $f : (\mathbb{Q}_+ \to A) \to B$ gives rise to an operation $\overline{A} \to B$. The equality axiom does not hold because we have changed the equality.
4. The λ-quantifier works as usual, except that it produces now partial operations instead of functions.
5. The universal quantifier is defined by $\forall_{x:\overline{A}}P(x) := \forall_{x:\mathbb{Q}_+\to A}(R(x) \to P(x))$.
6. The existential quantifier is defined by $\exists_{x:\overline{A}}P(x) := \exists_{x:\mathbb{Q}_+\to A}(R(x) \wedge P(x))$.

We define the relation $\overline{D} : \mathbb{Q}_+ \to \overline{A} \to \overline{A} \to H$ by

$$\overline{D}\epsilon := \exists_\alpha(\alpha < \epsilon \wedge \forall\beta(\beta < \alpha \to D(\epsilon - \alpha)(r\beta)(s\beta))).$$

Theorem 55 *The relation $\overline{D}$ induces a metric for $\overline{A}$.*

Proof: Omitted. Note that it does not suffice to prove all metric axioms, we also have to show that $\overline{D}$ is well-defined on $\overline{A}$.

Theorem 56 *The completion of a metric space is complete.*

Proof: Omitted.

Now, we define $\mathbb{R}$ as the completion of $\mathbb{Q}$ with respect to the metric $D\epsilon xy := |x-y| < \epsilon$.

A subset of a metric space A is just a predicate $A \to H$. A subset P of $\mathbb{R}$ is said to be bounded above iff

$$\exists_{x:\mathbb{R}} \forall_{y:\mathbb{R}} (Py \to y \leq x).$$

Such a real number x is called an upper bound for P. A real number b is called a supremum or least upper bound of P iff it is an upper bound and

$$\forall_\epsilon \exists_{x:\mathbb{R}} (Px \wedge x > b - \epsilon).$$

Dually, one defines lower bounds and infimums.

A set can have at most one supremum (infimum). Classically, every bounded set has a supremum. Constructively, this does not hold: imagine a set defined by a predicate which we cannot decide, in conjunction with the predicate $|x| \leq 1$. This subset is surely bounded, but we do not know whether the set is empty or the whole interval $[-1, 1]$.

Here is a constructive version of the lub-theorem.

Theorem 57 *Let P be a subset of $\mathbb{R}$. Then P has a supremum iff the following conditions hold.*

1. $\exists_{x:\mathbb{R}} Px$ *(P is inhabitated),*
2. *P is bounded above,*
3. *for all $x, y : \mathbb{R}$ such that $x < y$, we have y is an upper bound or $\exists_{a:\mathbb{R}} (Pa \wedge x < a)$.*

Proof: See [3], p 37.

At this point, we may redefine a metric as a function $d : A \to A \to \mathbb{R}$, satisfying the axioms

1. $d(x, y) \geq 0$,
2. $d(x, y) = 0 \leftrightarrow x = y$,
3. $d(x, y) = d(y, x)$,
4. $d(x, y) + d(y, z) \geq d(x, z)$,

where $x, y, z : A$. The old metric can be defined in terms of the new metric via $D(\epsilon, x, y) := d(x, y) < \epsilon$, and the new metric can be defined in terms of the old metric as the supremum of all ϵ such that $D(\epsilon, x, y)$.

Let A, B be metric spaces. A function $f : A \to B$ is called uniformly continuous iff there exists a function $\omega : \mathbb{Q}_+ \to \mathbb{Q}_+$, such that

$$\forall_\epsilon \forall_{x,y:A}(d_A(x,y) < \omega(\epsilon) \to d_B(fx, fy) < \epsilon).$$

Such a function is called a modulus of continuity.

The following theorem is useful to construct big classes of uniformly continuous functions.

Theorem 58 *Let A and B be metric spaces, B complete. Let $f : A \to B$ be uniformly continuous with modulus ω. Then there exists a unique function $\overline{f} : \overline{A} \to B$ which coincides with f on A and which is uniformly continuous. Moreover, $\omega/2$ is a modulus of continuity.*

Proof: See [3], p92.

A more important concept than boundedness is the concept of totally boundedness, defines as follows: a metric space A is said to be totally bounded iff there exists a function $a : \mathbb{N} \to A$ such that

$$\forall_\epsilon \exists_{n:\mathbb{N}} \forall_{x:A} \exists_{m:\mathbb{N}}(m < n \wedge d(am, x) < \epsilon).$$

The function a is called a net for A. A subset of A is bounded iff it is totally bounded as a space on its own.

Example 13 The interval $[0, 1]$ in $\mathbb{R}$ is bounded: enumerate all pairs (i, j) of natural numbers such that $i \leq j$ and $j \neq 0$, and define $a(n) := i/j$, where (i, j) is the n-th pair. For given ϵ, one selects n such that all pairs (i, j) with $j < \epsilon^{-1}$ having index less than n. This is possible because there are only finitely many of them.

Here is another constructive lub theorem.

Theorem 59 *Any totally bounded subset of $\mathbb{R}$ has a supremum.*

Proof: See [3], p 38.

Classically, a subset of a totally bounded set is totally bounded. Constructively, we already have seen a counter-example. However, we can say the following.

Theorem 60 *A subset P of a totally bounded space is totally bounded iff for all $x : A$ the subset $Qr := \exists_{y:A}(Py \wedge d(x, y) = r)$ has an infimum. A subset with such a property is called located.*

Proof: See [3], p 95.

Here are some theorems that show the usefulness of totally boundedness.

Theorem 61 *Let A be totally bounded and $f : A \to B$ be uniformly continuous. Then the image subset $(fA)(y) := \exists_{x:A} fx = y$ is totally bounded.*

Proof: If a is a net for A, then $f \circ a$ is a net for fA.

Corollary 10 *Let A be totally bounded and $f : A \to \mathbb{R}$ be uniformly continuous. Then fA has a supremum.*

Proof: Obvious from theorem 61 and theorem 59.

If $f : \mathbb{R} \to \mathbb{R}$ is uniformly continuous, and $\exists_{x,y:\mathbb{R}} fx \cdot fy < 0$, then one can show that 0 is the infimum of the image of $|f|$. Classically, 0 is also in the image; this is the mean value theorem. Constructively this theorem does not hold in general: let $u : \mathbb{R}$ be a real number for which we cannot decide whether $u = 0$, $u < 0$, or $u > 0$. Define $fx := \max(x - 1, (\min(x, u))$. Then f is uniformly continuous with modulus 1. Suppose that this function has a zero y. Then we could find information on the sign of u be deciding which of the parts of the disjunction $y < 2/3 \vee y > 1/3$ holds.

One can prove a mean value theorem using stronger assumptions, e.g. strong monotony $\forall_{x,y:\mathbb{R}}(x < y \to fx < fy)$.

References

[1] Barendregt, H. *The lambda calculus, its syntax and semantics.* Springer, 1981.

[2] Beeson, M. *Foundations of Constructive Mathematics.* Springer, 1985.

[3] Bishop, E. *Foundations of constructive analysis.* McGraw-Hill, 1967.

[4] Buchberger, B. *An Algorithm for Finding a Basis for the Residue Class Ring of a Zero-Dimensional Polynomial Ideal.* PhD thesis, Universitat Innsbruck, Institut fur Mathematik, 1965. German.

[5] Chang, B., and Lee, C. *Symbolic Logic and Mechanical Theorem Proving.* Academic Press, 1973.

[6] Collins, G. E. Quantifier elimination for the elementary theory of real closed fields by cylindrical algebraic decomposition. In *Lecture Notes In Computer Science* (1975), Springer, pp. 134–183. Vol. 33.

[7] Dedekind, R. *Stetigkeit und irrationale Zahlen.* Vieweg, 1872.

[8] Dummett, M. *Elements of Intuitionism.* Clarendon Press, 1977.

[9] Ebbinghaus, H., Flum, J., and Thomas, W. *Mathematical Logic.* Springer, 1983.

[10] H. Schwabhäuser and W. Szmielev and A. Tarski. *Metamathematische Methoden in der Geometrie.* Springer, 1983.

[11] Heyting, A. *Intuitionism.* North-Holland, 1966.

[12] Hilbert, D. *Grundlagen der Geometrie.* Teubner, 1977.

[13] Hilbert, D., and Bernays, P. *Grundlagen der Mathematik I und II.* Springer, 1970.

[14] Jebelean, T. Automatic theorem proving 1 and 2. Lectures at the university of Linz, winter 1998/99 and summer 1999.

[15] Jones, A., and Matijasevic, Y. Proof of the recursive unsolvability of hilbert's 10th problem. *Am. Math. Monthly 98* (1991), 689–709.

[16] Kleene, S. *Introduction to Metamathematics.* North-Holland, 1967.

[17] Manin, Y. *A course in mathematical logic.* Springer, 1977.

[18] Margaris, A. *First Order Mathematical Logic.* Xerox, 1967.

[19] Pfalzgraf, J., and Wang, D. *Automated practical reasoning.* Springer, 1995.

[20] Podniek. Hilbert's 10th problem. available via http://www.ltn.lv/ podnieks/gt4a.html.

[21] Renegar, J. On the computational complexity and geometry of the first-order theory of the reals i,ii,iii. *Journal of Symbolic Computation 13*, 3 (1992), 255–352.

[22] Rolletschek, H. Decidability and complexity classes. Tech. rep., RISC-Linz, Univ. Linz, A-4040 Linz. lecture notes.

[23] Rolletschek, H. Decidable logical theories. Tech. rep., RISC-Linz, Univ. Linz, A-4040 Linz. lecture notes.

[24] Rolletschek, H. Theory of computability. Tech. rep., RISC-Linz, Univ. Linz, A-4040 Linz. lecture notes.

[25] Shoenfield, J. *Mathematical Logic*. Addison-Wesley, 1973.

[26] Smullyan, R. *First order logic*. Springer, 1968.

[27] Stokkermans, K. An axiomatic frame for critical-pair-completion procedures. *Journal of Symbolic Computation 27*, 435–478.

[28] Tarski, A. *A Decision Method for Elementary Algebra and Geometry*, second ed. Univ. of California Press, Berkeley, 1951.

[29] Tarski, A. What is elementary geometry? In *Proceedings of an International Symposium on the Axiomatic Method, with Special Reference to Geometry and Physics* (1959), A. T. L. Henkin, P. Suppes, Ed., Studies in Logic and the Foundations of Mathematics, Amsterdam: North Holland, pp. 16–29.

[30] Troelstra, A., and van Dalen, D. *Constructivism in Mathematics*. North-Holland, 1988.

[31] Winkler, F. Rewriting in computer algebra and logic. Tech. rep., RISC-Linz, Univ. Linz, A-4040 Linz. lecture notes.

[32] Winkler, F. Theoretical computer science. Tech. rep., RISC-Linz, Univ. Linz, A-4040 Linz. lecture notes.

[33] Winkler, F. *The Church-Rosser property in computer algebra and special theorem proving*. PhD thesis, RISC-Linz, Univ. of Linz, 1984.