Mathematical Logic

Helmut Schwichtenberg

Mathematisches Institut der Universität München
Wintersemester 2009/2010

Contents

CHAPTER 1

Logic

The main subject of Mathematical Logic is mathematical proof. In this introductory chapter we deal with the basics of formalizing such proofs and, via normalization, analysing their structure. The system we pick for the representation of proofs is Gentzen's natural deduction from (1934). Our reasons for this choice are twofold. First, as the name says this is a *natural* notion of formal proof, which means that the way proofs are represented corresponds very much to the way a careful mathematician writing out all details of an argument would go anyway. Second, formal proofs in natural deduction are closely related (via the so-called Curry-Howard correspondence) to terms in typed lambda calculus. This provides us not only with a compact notation for logical derivations (which otherwise tend to become somewhat unmanagable tree-like structures), but also opens up a route to applying (in part 3) the computational techniques which underpin lambda calculus.

Apart from classical logic we will also deal with more constructive logics: minimal and intuitionistic logic. This will reveal some interesting aspects of proofs, e.g., that it is possible and useful to distinguish beween existential proofs that actually construct witnessing objects, and others that don't.

An essential point for Mathematical Logic is to fix a formal language to be used. We take implication $\to$ and the universal quantifier $\forall$ as basic. Then the logic rules correspond precisely to lambda calculus. The additional connectives: the existential quantifier $\exists$, disjunction $\vee$ and conjunction $\wedge$, can then be added either as rules or axiom schemes. It is "natural" to treat them as rules, and that is what we do here.

An underlying theme of this chapter is to bring out the constructive content of logic, particularly in regard to the relationship between minimal and classical logic. For us the latter is most appropriately viewed as a subsystem of the former.

1.1. Natural Deduction

Rules come in pairs: we have an introduction and an elimination rule for each of the logical connectives. The resulting system is called *minimal logic*;

it was introduced by Kolmogorov (1932), Gentzen (1934) and Johansson (1937). Notice that no negation is yet present. If we go on and require *ex-falso-quodlibet* for the nullary propositional symbol $\bot$ ("falsum") we can embed *intuitionistic logic* with negation as $A \to \bot$. To embed classical logic, we need to go further and add as an axiom schema the principle of *indirect proof*, also called *stability* ($\forall_{\vec{x}}(\neg\neg R\vec{x} \to R\vec{x}$) for relation symbols R), but then it is appropriate to restrict to the language based on $\to$, $\forall$, $\bot$ and $\wedge$. The reason for this restriction is that we can neither prove $\neg\neg\exists_x A \to \exists_x A$ nor $\neg\neg(A \vee B) \to A \vee B$, for there are countermodels to both (the former is Markov's scheme). However, we can prove them for the classical existential quantifier and disjunction defined by $\neg\forall_x\neg A$ and $\neg A \to \neg B \to \bot$. Thus we need to make a distinction between two kinds of "exists" and two kinds of "or": the classical ones are "weak" and the non-classical ones "strong" since they have constructive content. In situations where both kinds occur together we must mark the distinction, and we shall do this by writing a tilde above the weak disjunction and existence symbols thus $\tilde{\vee}, \tilde{\exists}$. Of course, in a classical context this distinction does not arise and the tilde is not necessary.

1.1.1. Terms and formulas. Let a countably infinite set $\{ v_i \mid i \in \mathbb{N} \}$ of *variables* be given; they will be denoted by x, y, z. A first order language $\mathcal{L}$ then is determined by its *signature*, which is to mean the following.

(i) For every natural number $n \geq 0$ a (possible empty) set of n-ary *relation symbols* (or *predicate symbols*). 0-ary relation symbols are called *propositional symbols*. $\bot$ (read "falsum") is required as a fixed propositional symbol. The language will *not*, unless stated otherwise, contain $=$ as a primitive. Binary relation symbols can be marked as *infix*.

(ii) For every natural number $n \geq 0$ a (possible empty) set of n-ary *function symbols*. 0-ary function symbols are called *constants*. Binary function sysmbols can also be marked as infix.

We assume that all these sets of variables, relation and function symbols are disjoint. $\mathcal{L}$ is kept fixed and will only be mentioned when necessary.

Terms are inductively defined as follows.

(i) Every variable is a term.

(ii) Every constant is a term.

(iii) If $t_1, \ldots, t_n$ are terms and f is an n-ary function symbol with $n \geq 1$, then $f(t_1, \ldots, t_n)$ is a term. (If r, s are terms and $\circ$ is a binary function symbol, then $(r \circ s)$ is a term.)

From terms one constructs *prime formulas*, also called *atomic formulas* or just *atoms*: If $t_1, \ldots, t_n$ are terms and R is an n-ary relation symbol, then $R(t_1, \ldots, t_n)$ is a prime formula. (If r, s are terms and $\sim$ is a binary relation symbol, then $(r \sim s)$ is a prime formula.)

Formulas are inductively defined from prime formulas by

(i) Every prime formula is a formula.
(ii) If A and B are formulas, then so are $(A \to B)$ ("if A then B"), $(A \wedge B)$ ("A and B") and $(A \vee B)$ ("A or B").
(iii) If A is a formula and x is a variable, then $\forall_x A$ ("A holds for all x") and $\exists_x A$ ("there is an x such that A") are formulas.

Negation is defined by

$$\neg A := (A \to \bot).$$

NOTATION. In writing formulas we save on parentheses by assuming that $\forall, \exists, \neg$ bind more strongly than $\wedge, \vee$, and that in turn $\wedge, \vee$ bind more strongly than $\to, \leftrightarrow$ (where $A \leftrightarrow B$ abbreviates $(A \to B) \wedge (B \to A)$). Outermost parentheses can be dropped. Thus $A \wedge \neg B \to C$ is read as $((A \wedge (\neg B)) \to C)$. In the case of iterated implications we use the short notation

$$A_1 \to A_2 \to \cdots \to A_{n-1} \to A_n \text{ for } A_1 \to (A_2 \to \cdots \to (A_{n-1} \to A_n) \dots).$$

We also occasionally save on parentheses by writing for instance $Rxyz$, $Rt_0t_1t_2$ instead of $R(x, y, z)$, $R(t_0, t_1, t_2)$, where R is some predicate symbol. Similarly for a unary function symbol with a (typographically) simple argument, so fx for $f(x)$, etc. In this case no confusion will arise. But readability requires that we write in full $R(fx, gy, hz)$, instead of $Rfxgyhz$.

We shall often need to do induction on the height, denoted $|A|$, of formulas A. This is defined as follows: $|P| = 0$ for atoms P, $|A \circ B| = \max(|A|, |B|) + 1$ for binary operators $\circ$ (i.e., $\to, \wedge, \vee$) and $|\circ A| = |A| + 1$ for unary operators $\circ$ (i.e., $\forall_x$, $\exists_x$).

1.1.2. Substitution, free and bound variables. Expressions $\mathcal{E}, \mathcal{E}'$ which differ only in the names of bound variables will be regarded as identical. This is sometimes expressed by saying that $\mathcal{E}$ and $\mathcal{E}'$ are α-equivalent. In other words, we are only interested in expressions "modulo renaming of bound variables". There are methods of finding unique representatives for such expressions, for example the name-free terms of de Bruijn (1972). For the human reader such representations are less convenient, so we shall stick to the use of bound variables.

In the definition of "substitution of expression $\mathcal{E}'$ for variable x in expression $\mathcal{E}$", either one requires that *no* variable free in $\mathcal{E}'$ becomes bound by a variable-binding operator in $\mathcal{E}$, when the free occurrences of x are replaced by $\mathcal{E}'$ (also expressed by saying that there must be no "clashes of variables"), "*$\mathcal{E}'$ is free for x in $\mathcal{E}$*", or the substitution operation is taken to involve a systematic renaming operation for the bound variables, avoiding clashes. Having stated that we are only interested in expressions modulo

renaming bound variables, we can without loss of generality assume that substitution is always possible.

Also, it is never a real restriction to assume that distinct quantifier occurrences are followed by distinct variables, and that the sets of bound and free variables of a formula are disjoint.

NOTATION. "FV" is used for the (set of) free variables of an expression; so $\mathrm{FV}(r)$ is the set of variables free in the term r, $\mathrm{FV}(A)$ the set of variables free in formula A etc. A formula A is said to be *closed* if $\mathrm{FV}(A) = \emptyset$.

$\mathcal{E}[x := r]$ denotes the result of substituting the term r for the variable x in the expression $\mathcal{E}$. Similarly, $\mathcal{E}[\vec{x} := \vec{r}]$ is the result of *simultaneously* substituting the terms $\vec{r} = r_1, \ldots, r_n$ for the variables $\vec{x} = x_1, \ldots, x_n$, respectively.

In a given context we shall adopt the following convention. Once a formula has been introduced as $A(x)$, i.e., A with a designated variable x, we write $A(r)$ for $A[x := r]$, and similarly with more variables. □

1.1.3. Subformulas. Unless stated otherwise, the notion of *subformula* will be that defined by Gentzen.

DEFINITION. (Gentzen) subformulas of A are defined by

(a) A is a subformula of A;
(b) if $B \circ C$ is a subformula of A then so are B, C, for $\circ = \rightarrow, \wedge, \vee$;
(c) if $\forall_x B(x)$ or $\exists_x B(x)$ is a subformula of A, then so is $B(r)$.

DEFINITION. The notions of *positive, negative, strictly positive* subformula are defined in a similar style:

(a) A is a positive and a strictly positive subformula of itself;
(b) if $B \wedge C$ or $B \vee C$ is a positive (negative, strictly positive) subformula of A, then so are B, C;
(c) if $\forall_x B(x)$ or $\exists_x B(x)$ is a positive (negative, strictly positive) subformula of A, then so is $B(r)$;
(d) if $B \rightarrow C$ is a positive (negative) subformula of A, then B is a negative (positive) subformula of A, and C is a positive (negative) subformula of A;
(e) if $B \rightarrow C$ is a strictly positive subformula of A, then so is C.

A strictly positive subformula of A is also called a *strictly positive part (s.p.p.)* of A. Note that the set of subformulas of A is the union of the positive and negative subformulas of A.

EXAMPLE. $(P \rightarrow Q) \rightarrow R \wedge \forall_x S(x)$ has as s.p.p.'s the whole formula, $R \wedge \forall_x S(x)$, R, $\forall_x S(x)$, $S(r)$. The positive subformulas are the s.p.p.'s and in addition P; the negative subformulas are $P \rightarrow Q$, Q.

1.1.4. Examples of derivations. To motivate the rules for natural deduction, let us start with informal proofs of some simple logical facts.

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

Informal proof. Assume $A \to B \to C$. To show: $(A \to B) \to A \to C$. So assume $A \to B$. To show: $A \to C$. So finally assume A. To show: C. Using the third assumption twice we have $B \to C$ by the first assumption, and B by the second assumption. From $B \to C$ and B we then obtain C. Then $A \to C$, cancelling the assumption on A; $(A \to B) \to A \to C$ cancelling the second assumption; and the result follows by cancelling the first assumption. □

$$\forall_x(A \to B) \to A \to \forall_x B, \quad \text{if } x \notin \mathrm{FV}(A).$$

Informal proof. Assume $\forall_x(A \to B)$. To show: $A \to \forall_x B$. So assume A. To show: $\forall_x B$. Let x be arbitrary; note that we have not made any assumptions on x. To show: B. We have $A \to B$ by the first assumption. Hence also B by the second assumption. Hence $\forall_x B$. Hence $A \to \forall_x B$, cancelling the second assumption. Hence the result, cancelling the first assumption. □

A characteristic feature of these proofs is that assumptions are introduced and eliminated again. At any point in time during the proof the free or "open" assumptions are known, but as the proof progresses, free assumptions may become cancelled or "closed" because of the implies-introduction rule.

We reserve the word *proof* for the informal level; a formal representation of a proof will be called a *derivation*.

An intuitive way to communicate derivations is to view them as labelled trees each node of which denotes a rule application. The labels of the inner nodes are the formulas derived as conclusions at those points, and the labels of the leaves are formulas or terms. The labels of the nodes immediately above a node k are the *premises* of the rule application. At the root of the tree we have the conclusion (or end formula) of the whole derivation. In natural deduction systems one works with *assumptions* at leaves of the tree; they can be either *open* or *closed* (cancelled). Any of these assumptions carries a *marker*. As markers we use *assumption variables* denoted $u, v, w, u_0, u_1, \ldots$. The variables of the language previously introduced will now often be called *object variables*, to distinguish them from assumption variables. If at a node below an assumption the dependency on this assumption is removed (it becomes closed) we record this by writing down the assumption variable. Since the same assumption may be used more than once (this was the case in the first example above), the assumption marked with u (written $u\colon A$) may appear many times. Of course we insist that distinct assumption formulas must have distinct markers. An inner node of

the tree is understood as the result of passing from premises to the conclusion of a given rule. The label of the node then contains, in addition to the conclusion, also the name of the rule. In some cases the rule binds or closes or cancels an assumption variable u (and hence removes the dependency of all assumptions $u\colon A$ thus marked). An application of the $\forall$-introduction rule similarly binds an object variable x (and hence removes the dependency on x). In both cases the bound assumption or object variable is added to the label of the node.

DEFINITION. A formula A is called *derivable* (in *minimal logic*), written $\vdash A$, if there is a derivation of A (without free assumptions) using the natural deduction rules. A formula B is called derivable from assumptions $A_1, \dots, A_n$, if there is a derivation of B with free assumptions among $A_1, \dots, A_n$. Let Γ be a (finite or infinite) set of formulas. We write $\Gamma \vdash B$ if the formula B is derivable from finitely many assumptions $A_1, \dots, A_n \in \Gamma$.

We now formulate the rules of natural deduction.

1.1.5. Introduction and elimination rules for $\to$ and $\forall$. First we have an assumption rule, allowing to write down an arbitrary formula A together with a marker u:

$$u\colon A \qquad \text{assumption.}$$

The other rules of natural deduction split into introduction rules (I-rules for short) and elimination rules (E-rules) for the logical connectives which, for the time being, are just $\to$ and $\forall$. For implication $\to$ there is an introduction rule $\to^+$ and an elimination rule $\to^-$ also called *modus ponens*. The left premise $A \to B$ in $\to^-$ is called the *major* (or *main*) premise, and the right premise A the *minor* (or *side*) premise. Note that with an application of the $\to^+$-rule *all* assumptions above it marked with $u\colon A$ are cancelled (which is denoted by putting square brackets around these assumptions), and the u then gets written alongside. There may of course be other uncancelled assumptions $v\colon A$ of the same formula A, which may get cancelled at a later stage.

$$\frac{\begin{array}{c}[u\colon A]\\ \vert\, M\\ B\end{array}}{A \to B}\to^+ u \qquad \frac{\vert\, M \qquad \vert\, N}{A \to B \qquad A}\to^- \\ B$$

For the universal quantifier $\forall$ there is an introduction rule $\forall^+$ (again marked, but now with the bound variable x) and an elimination rule $\forall^-$ whose right premise is the term r to be substituted. The rule $\forall^+ x$ with conclusion $\forall_x A$ is subject to the following *(Eigen-)variable condition*: the derivation M of

the premise A should not contain any open assumption having x as a free variable.

$$\frac{\begin{array}{c}|\,M\\ A\end{array}}{\forall_x A}\,\forall^+x \qquad \frac{\begin{array}{cc}|\,M & \\ \forall_x A(x) & r\end{array}}{A(r)}\,\forall^-$$

We now give derivations of the two example formulas treated informally above. Since in many cases the rule used is determined by the conclusion, we suppress in such cases the name of the rule.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{u\colon A\to B\to C \qquad w\colon A}{B\to C} \qquad \dfrac{v\colon A\to B \qquad w\colon A}{B}}{C}\,{\to^+}w}{A\to C}\,{\to^+}v}{(A\to B)\to A\to C}\,{\to^+}u}{(A\to B\to C)\to(A\to B)\to A\to C}$$

$$\dfrac{\dfrac{\dfrac{\dfrac{u\colon \forall_x(A\to B) \qquad x}{A\to B} \qquad v\colon A}{B}\,\forall^+x}{\forall_x B}\,{\to^+}v}{A\to\forall_x B}\,{\to^+}u}{\forall_x(A\to B)\to A\to\forall_x B}$$

Note that the variable condition is satisfied: x is not free in A (and also not free in $\forall_x(A\to B)$).

1.1.6. Properties of negation. Recall that negation is defined by $\neg A := (A\to\bot)$. The following can easily be derived.

$$A\to\neg\neg A,$$
$$\neg\neg\neg A\to\neg A.$$

However, $\neg\neg A\to A$ is in general *not* derivable (without stability – we will come back to this later on).

Lemma. *The following are derivable.*

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

$$\neg\neg(A\to B)\to\neg\neg A\to\neg\neg B,$$
$$(\bot\to B)\to(\neg\neg A\to\neg\neg B)\to\neg\neg(A\to B),$$

$$\neg\neg\forall_x A\to\forall_x\neg\neg A.$$

Derivations are left as an exercise.

1.1.7. Introduction and elimination rules for disjunction $\vee$, conjunction $\wedge$ and existence $\exists$. For disjunction the introduction and elimination rules are

$$\frac{\begin{array}{c}|\,M\\ A\end{array}}{A\vee B}\vee_0^+ \qquad \frac{\begin{array}{c}|\,M\\ B\end{array}}{A\vee B}\vee_1^+ \qquad \frac{\begin{array}{ccc}&[u\colon A]&[v\colon B]\\ |\,M&|\,N&|\,K\\ A\vee B&C&C\end{array}}{C}\vee^- u,v$$

For conjunction we have

$$\frac{\begin{array}{cc}|\,M&|\,N\\ A&B\end{array}}{A\wedge B}\wedge^+ \qquad \frac{\begin{array}{ccc}&[u\colon A]&[v\colon B]\\ |\,M&&|\,N\\ A\wedge B&&C\end{array}}{C}\wedge^- u,v$$

and for the existential quantifier

$$\frac{\begin{array}{cc}&|\,M\\ r&A(r)\end{array}}{\exists_x A(x)}\exists^+ \qquad \frac{\begin{array}{ccc}&[u\colon A]\\ |\,M&|\,N\\ \exists_x A&B\end{array}}{B}\exists^- x,u\ \text{(var.cond.)}$$

Similar to $\forall^+ x$ the rule $\exists^- x,u$ is subject to an *(Eigen-)variable condition*: in the derivation N the variable x (i) should not occur free in the formula of any open assumption other than $u\colon A$, and (ii) should not occur free in B.

Again, in each of the elimination rules $\vee^-$, $\wedge^-$ and $\exists^-$ the left premise is called *major* (or *main*) premise, and the right premise is called the *minor* (or *side*) premise.

It is easy to see that for each of the connectives $\vee$, $\wedge$, $\exists$ the rules and the following axioms are equivalent over minimal logic; this is left as an exercise. For disjunction the introduction and elimination axioms are

$$\begin{aligned}&\vee_0^+\colon A\to A\vee B,\\ &\vee_1^+\colon B\to A\vee B,\\ &\vee^-\colon A\vee B\to(A\to C)\to(B\to C)\to C.\end{aligned}$$

For conjunction we have

$$\wedge^+\colon A\to B\to A\wedge B,\qquad \wedge^-\colon A\wedge B\to(A\to B\to C)\to C$$

and for the existential quantifier

$$\exists^+\colon A\to\exists_x A,\qquad \exists^-\colon\exists_x A\to\forall_x(A\to B)\to B\quad(x\notin \mathrm{FV}(B)).$$

REMARK. All these axioms can be seen as special cases of a general schema, that of an *inductively defined predicate*, which is defined by some introduction rules and one elimination rule. Later we will study this kind of definition in full generality.

We collect some easy facts about derivability; $B \leftarrow A$ means $A \to B$.

LEMMA. *The following are derivable.*

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

$$(A \vee B \to C) \leftrightarrow (A \to C) \wedge (B \to C),$$
$$(A \to B \vee C) \leftarrow (A \to B) \vee (A \to C),$$

$$(\forall_x A \to B) \leftarrow \exists_x (A \to B) \quad \textit{if } x \notin \mathrm{FV}(B),$$
$$(A \to \forall_x B) \leftrightarrow \forall_x (A \to B) \quad \textit{if } x \notin \mathrm{FV}(A),$$

$$(\exists_x A \to B) \leftrightarrow \forall_x (A \to B) \quad \textit{if } x \notin \mathrm{FV}(B),$$
$$(A \to \exists_x B) \leftarrow \exists_x (A \to B) \quad \textit{if } x \notin \mathrm{FV}(A).$$

PROOF. A derivation of the final formula is

$$\dfrac{\dfrac{u\colon \exists_x(A \to B) \qquad \dfrac{\dfrac{}{x} \qquad \dfrac{w\colon A \to B \qquad v\colon A}{B}}{\exists_x B}\,\exists^- x, w}{\exists_x B}}{\dfrac{A \to \exists_x B}{\exists_x(A \to B) \to A \to \exists_x B}\,{\to^+} u}\,{\to^+} v$$

The variable condition for $\exists^-$ is satisfied since the variable x (i) is not free in the formula A of the open assumption $v\colon A$, and (ii) is not free in $\exists_x B$. The rest of the proof is left as an exercise. □

As already mentioned, we distinguish between two kinds of "exists" and two kinds of "or": the "weak" or classical ones and the "strong" or non-classical ones, with constructive content. In the present context both kinds occur together and hence we must mark the distinction; we shall do this by writing a tilde above the weak disjunction and existence symbols thus

$$A \mathbin{\tilde{\vee}} B := \neg A \to \neg B \to \bot, \qquad \tilde{\exists}_x A := \neg \forall_x \neg A.$$

One can show easily that these weak variants of disjunction and the existential quantifier are no stronger than the proper ones (in fact, they are weaker):

$$A \vee B \to A \mathbin{\tilde{\vee}} B, \qquad \exists_x A \to \tilde{\exists}_x A.$$

This can be seen easily by putting $C := \bot$ in $\vee^-$ and $B := \bot$ in $\exists^-$.

1.1.8. Intuitionistic and classical derivability. In the definition of derivability in 1.1.4 falsity $\bot$ plays no role. We may change this and require *ex-falso-quodlibet* axioms, of the form

$$\forall_{\vec{x}}(\bot \to R\vec{x}\,)$$

with R a relation symbol distinct from $\bot$. Let Efq denote the set of all such axioms. A formula A is called *intuitionistically derivable*, written $\vdash_i A$, if $\mathrm{Efq} \vdash A$. We write $\Gamma \vdash_i B$ for $\Gamma \cup \mathrm{Efq} \vdash B$.

We may even go further and require *stability* axioms, of the form

$$\forall_{\vec{x}}(\neg\neg R\vec{x} \to R\vec{x}\,)$$

with R again a relation symbol distinct from $\bot$. Let Stab denote the set of all these axioms. A formula A is called *classically derivable*, written $\vdash_c A$, if $\mathrm{Stab} \vdash A$. We write $\Gamma \vdash_c B$ for $\Gamma \cup \mathrm{Stab} \vdash B$.

It is easy to see that intuitionistically (i.e., from Efq) we can derive $\bot \to A$ for an *arbitrary* formula A, using the introduction rules for the connectives. A similar generalization of the stability axioms is only possible for formulas in the language not involving $\vee, \exists$. However, it is still possible to use the substitutes $\tilde{\vee}$ and $\tilde{\exists}$.

THEOREM (Stability, or principle of indirect proof).
(a) $\vdash (\neg\neg A \to A) \to (\neg\neg B \to B) \to \neg\neg(A \wedge B) \to A \wedge B$.
(b) $\vdash (\neg\neg B \to B) \to \neg\neg(A \to B) \to A \to B$.
(c) $\vdash (\neg\neg A \to A) \to \neg\neg\forall_x A \to A$.
(d) $\vdash_c \neg\neg A \to A$ *for every formula* A *without* $\vee, \exists$.

PROOF. (a) is left as an exercise. (b). For simplicity, in the derivation to be constructed we leave out applications of $\to^+$ at the end.

$$\dfrac{u\colon \neg\neg B \to B \qquad \dfrac{\dfrac{v\colon \neg\neg(A \to B) \qquad \dfrac{\dfrac{u_1\colon \neg B \qquad \dfrac{u_2\colon A \to B \qquad w\colon A}{B}}{\bot}\to^+ u_2}{\neg(A \to B)}}{\bot}\to^+ u_1}{\neg\neg B}}{B}$$

(c).

$$\dfrac{u\colon \neg\neg A \to A \qquad \dfrac{\dfrac{v\colon \neg\neg\forall_x A \qquad \dfrac{\dfrac{u_1\colon \neg A \qquad \dfrac{u_2\colon \forall_x A \qquad x}{A}}{\bot}\to^+ u_2}{\neg\forall_x A}}{\bot}\to^+ u_1}{\neg\neg A}}{A}$$

(d). Induction on A. The case $R\vec{t}$ with R distinct from $\bot$ is given by Stab. In the case $\bot$ the desired derivation is

$$\dfrac{v\colon (\bot\to\bot)\to\bot \qquad \dfrac{u\colon \bot}{\bot\to\bot}\to^+ u}{\bot}$$

In the cases $A\wedge B$, $A\to B$ and $\forall_x A$ use (a), (b) and (c), respectively. □

Using stability we can prove some well-known facts about the interaction of weak disjunction and the weak existential quantifier with implication. We first prove a more refined claim, stating to what extent we need to go beyond minimal logic.

Lemma. *The following are derivable.*

$$(\tilde\exists_x A\to B)\to\forall_x(A\to B) \quad \text{if } x\notin \mathrm{FV}(B), \tag{1.1}$$

$$(\neg\neg B\to B)\to \quad \forall_x(A\to B)\to\tilde\exists_x A\to B \quad \text{if } x\notin \mathrm{FV}(B), \tag{1.2}$$

$$(\bot\to B[x{:=}c])\to(A\to\tilde\exists_x B)\to\tilde\exists_x(A\to B) \quad \text{if } x\notin \mathrm{FV}(A), \tag{1.3}$$

$$\tilde\exists_x(A\to B)\to A\to\tilde\exists_x B \quad \text{if } x\notin \mathrm{FV}(A). \tag{1.4}$$

The last two items can also be seen as simplifying a weakly existentially quantified implication whose premise does not contain the quantified variable. In case the conclusion does not contain the quantified variable we have

$$(\neg\neg B\to B)\to \quad \tilde\exists_x(A\to B)\to\forall_x A\to B \quad \text{if } x\notin \mathrm{FV}(B), \tag{1.5}$$

$$\forall_x(\neg\neg A\to A)\to(\forall_x A\to B)\to\tilde\exists_x(A\to B) \quad \text{if } x\notin \mathrm{FV}(B). \tag{1.6}$$

Proof. (1.1)

$$\dfrac{\tilde\exists_x A\to B \qquad \dfrac{\dfrac{u_1\colon \forall_x\neg A \qquad x}{\neg A} \qquad A}{\bot}\to^+ u_1}{\neg\forall_x\neg A}$$

(1.2)

$$\dfrac{\neg\neg B\to B \qquad \dfrac{\dfrac{\dfrac{u_2\colon \neg B \qquad \dfrac{\dfrac{\forall_x(A\to B) \qquad x}{A\to B} \qquad u_1\colon A}{B}}{\bot}\to^+ u_1}{\neg A}}{\forall_x\neg A}}{\neg\forall_x\neg A}$$

(1.3) Writing B_0 for $B[x{:=}c]$ we have

$$\dfrac{\dfrac{\forall_x\neg(A\to B)\quad c}{\neg(A\to B_0)}\qquad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\forall_x\neg(A\to B)\quad x}{\neg(A\to B)}\qquad \dfrac{u_1\colon B}{A\to B}}{\bot}}{\neg B}\to^+u_1}{\forall_x\neg B}\qquad \dfrac{\dfrac{A\to\tilde\exists_x B\quad u_2\colon A}{\tilde\exists_x B}\qquad \dfrac{\bot\to B_0\qquad \bot}{B_0}}{A\to B_0}\to^+u_2}{\bot}$$

(1.4)

$$\dfrac{\tilde\exists_x(A\to B)\qquad \dfrac{\dfrac{\dfrac{\forall_x\neg B\qquad x}{\neg B}\qquad \dfrac{u_1\colon A\to B\qquad A}{B}}{\bot}}{\neg(A\to B)}\to^+u_1}{\forall_x\neg(A\to B)}}{\bot}$$

(1.5)

$$\dfrac{\neg\neg B\to B\qquad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\forall_x A\qquad x}{A}\qquad u_1\colon A\to B}{B}\qquad u_2\colon\neg B}{\bot}}{\neg(A\to B)}\to^+u_1}{\forall_x\neg(A\to B)}\qquad \dfrac{\tilde\exists_x(A\to B)}{\bot}}{\neg\neg B}\to^+u_2}{B}$$

(1.6) We derive $\forall_x(\bot\to A)\to(\forall_x A\to B)\to\forall_x\neg(A\to B)\to\neg\neg A$. Writing Ax, Ay for $A(x), A(y)$ we have

$$\dfrac{\dfrac{\forall_x\neg(Ax\to B)\quad x}{\neg(Ax\to B)}\qquad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\forall_y(\bot\to Ay)\quad y}{\bot\to Ay}\qquad \dfrac{u_1\colon\neg Ax\quad u_2\colon Ax}{\bot}}{Ay}}{\forall_y Ay}\qquad \dfrac{\forall_x Ax\to B}{B}}{Ax\to B}\to^+u_2}{\bot}}{\neg\neg Ax}\to^+u_1$$

Using this derivation M we obtain

$$\dfrac{\dfrac{\forall_x\neg(Ax\to B)\quad x}{\neg(Ax\to B)}\qquad \dfrac{\forall_x Ax\to B\qquad \dfrac{\dfrac{\dfrac{\forall_x(\neg\neg Ax\to Ax)\quad x}{\neg\neg Ax\to Ax}\qquad \mid M\qquad \neg\neg Ax}{Ax}}{\forall_x Ax}}{B}}{Ax\to B}}{\bot}$$

Since clearly $\vdash(\neg\neg A\to A)\to\bot\to A$ the claim follows. □

Remark. An immediate consequence of (1.6) is the classical derivability of the "drinker formula" $\tilde\exists_x(Px\to\forall_x Px)$, to be read "in every non-empty bar there is a person such that, if this person drinks, then everybody drinks". To see this let $A:=Px$ and $B:=\forall_x Px$ in (1.6).

Corollary.

$\vdash_c(\tilde\exists_x A\to B)\leftrightarrow\forall_x(A\to B)$ *if* $x\notin\mathrm{FV}(B)$ *and* B *without* $\vee,\exists$,

$\vdash_i(A\to\tilde\exists_x B)\leftrightarrow\tilde\exists_x(A\to B)$ *if* $x\notin\mathrm{FV}(A)$,

$\vdash_c\tilde\exists_x(A\to B)\leftrightarrow(\forall_x A\to B)$ *if* $x\notin\mathrm{FV}(B)$ *and* A,B *without* $\vee,\exists$.

There is a similar lemma on weak disjunction:

Lemma. *The following are derivable.*

$$\begin{aligned}
&(A\,\tilde\vee\,B\to C)\to(A\to C)\wedge(B\to C),\\
(\neg\neg C\to C)\to\;&(A\to C)\to(B\to C)\to A\,\tilde\vee\,B\to C,\\
(\bot\to B)\to\;&(A\to B\,\tilde\vee\,C)\to(A\to B)\,\tilde\vee\,(A\to C),\\
&(A\to B)\,\tilde\vee\,(A\to C)\to A\to B\,\tilde\vee\,C,\\
(\neg\neg C\to C)\to\;&(A\to C)\,\tilde\vee\,(B\to C)\to A\to B\to C,\\
(\bot\to C)\to\;&(A\to B\to C)\to(A\to C)\,\tilde\vee\,(B\to C).
\end{aligned}$$

Proof. The derivation of the final formula is

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{A\to B\to C\quad u_1\colon A}{B\to C}\qquad u_2\colon B}{C}}{A\to C}\to^+u_1}{\neg(A\to C)}\qquad\bot}{\bot\to C}}{C}}{B\to C}\to^+u_2\qquad\neg(B\to C)}{\bot}$$

The other derivations are similar to the ones above, if one views $\tilde{\exists}$ as an infinitary version of $\tilde{\vee}$. □

COROLLARY.

$$\vdash_c (A \mathbin{\tilde{\vee}} B \to C) \leftrightarrow (A \to C) \wedge (B \to C) \quad \textit{for } C \textit{ without } \vee, \exists,$$
$$\vdash_i (A \to B \mathbin{\tilde{\vee}} C) \leftrightarrow (A \to B) \mathbin{\tilde{\vee}} (A \to C),$$
$$\vdash_c (A \to C) \mathbin{\tilde{\vee}} (B \to C) \leftrightarrow (A \to B \to C) \quad \textit{for } C \textit{ without } \vee, \exists.$$

REMARK. It is easy to see that weak disjunction and the weak existential quantifier satisfy the same axioms as the strong variants, if one restricts the conclusion of the elimination axioms to formulas without $\vee, \exists$. In fact, we have

$$\vdash A \to A \mathbin{\tilde{\vee}} B, \quad \vdash B \to A \mathbin{\tilde{\vee}} B,$$
$$\vdash_c A \mathbin{\tilde{\vee}} B \to (A \to C) \to (B \to C) \to C \quad (C \text{ without } \vee, \exists),$$
$$\vdash A \to \tilde{\exists}_x A,$$
$$\vdash_c \tilde{\exists}_x A \to \forall_x (A \to B) \to B \quad (x \notin \mathrm{FV}(B),\ B \text{ without } \vee, \exists).$$

The derivations are left as exercises.

1.1.9. Gödel-Gentzen translation. Classical derivability $\Gamma \vdash_c B$ was defined in 1.1.8 by $\Gamma \cup \mathrm{Stab} \vdash B$. This embedding of classical logic into minimal logic can be expressed in a somewhat different and very explicit form, namely as a syntactic translation $A \mapsto A^g$ of formulas such that A is derivable in classical logic if and only if its translation A^g is derivable in minimal logic.

DEFINITION (Gödel-Gentzen translation A^g).

$$\begin{aligned}
(R\vec{t}\,)^g &:= \neg\neg R\vec{t} \quad \text{for } R \text{ distinct from } \bot,\\
\bot^g &:= \bot,\\
(A \vee B)^g &:= A^g \mathbin{\tilde{\vee}} B^g,\\
(\exists_x A)^g &:= \tilde{\exists}_x A^g,\\
(A \circ B)^g &:= A^g \circ B^g \quad \text{for } \circ = \to, \wedge,\\
(\forall_x A)^g &:= \forall_x A^g.
\end{aligned}$$

LEMMA. $\vdash \neg\neg A^g \to A^g$.

PROOF. Induction on A.

Case $R\vec{t}$ with R distinct from $\bot$. We must show $\neg\neg\neg\neg R\vec{t} \to \neg\neg R\vec{t}$, which is a special case of $\vdash \neg\neg\neg B \to \neg B$.

Case $\bot$. Use $\vdash \neg\neg\bot \to \bot$.

Case $A \vee B$. We must show $\vdash \neg\neg(A^g \mathbin{\tilde\vee} B^g) \to A^g \mathbin{\tilde\vee} B^g$, which is a special case of $\vdash \neg\neg(\neg C \to \neg D \to \bot) \to \neg C \to \neg D \to \bot$:

$$\dfrac{\neg\neg(\neg C \to \neg D \to \bot) \qquad \dfrac{\dfrac{\dfrac{u_1 \colon \neg C \to \neg D \to \bot \qquad \neg C}{\neg D \to \bot} \qquad \neg D}{\bot}}{\neg(\neg C \to \neg D \to \bot)} \to^+ u_1}{\bot}$$

Case $\exists_x A$. In this case we must show $\vdash \neg\neg\tilde\exists_x A^g \to \tilde\exists_x A^g$, but this is a special case of $\vdash \neg\neg\neg B \to \neg B$, because $\tilde\exists_x A^g$ is the negation $\neg\forall_x \neg A^g$.

Case $A \wedge B$. We must show $\vdash \neg\neg(A^g \wedge B^g) \to A^g \wedge B^g$. By induction hypothesis $\vdash \neg\neg A^g \to A^g$ and $\vdash \neg\neg B^g \to B^g$. Now use part (a) of the stability theorem in 1.1.8.

The cases $A \to B$ and $\forall_x A$ are similar, using parts (b) and (c) of the stability theorem instead. □

THEOREM. (a) $\Gamma \vdash_c A$ *implies* $\Gamma^g \vdash A^g$.
(b) $\Gamma^g \vdash A^g$ *implies* $\Gamma \vdash_c A$ *for* Γ, A *without* $\vee, \exists$.

PROOF. (a). We use induction on $\Gamma \vdash_c A$. For a stability axiom $\forall_{\vec x}(\neg\neg R\vec x \to R\vec x)$ we must derive $\forall_{\vec x}(\neg\neg\neg\neg R\vec x \to \neg\neg R\vec x)$, which is easy (as above). For the rules $\to^+$, $\to^-$, $\forall^+$, $\forall^-$, $\wedge^+$ and $\wedge^-$ the claim follows immediately from the induction hypothesis, using the same rule again. This works because the Gödel-Gentzen translation acts as a homomorphism for these connectives. For the rules $\vee_i^+$, $\vee^-$, $\exists^+$ and $\exists^-$ the claim follows from the induction hypothesis and the remark at the end of 1.1.8. For example, in case $\exists^-$ the induction hypothesis gives

$$\begin{array}{c} \\ \mid M \\ \tilde\exists_x A^g \end{array} \quad \text{and} \quad \begin{array}{c} u \colon A^g \\ \mid N \\ B^g \end{array}$$

with $x \notin \mathrm{FV}(B^g)$. Now use $\vdash (\neg\neg B^g \to B^g) \to \tilde\exists_x A^g \to \forall_x(A^g \to B^g) \to B^g$. Its premise $\neg\neg B^g \to B^g$ is derivable by the lemma above.

(b). First note that $\vdash_c (B \leftrightarrow B^g)$ if B is without $\vee, \exists$. Now assume that Γ, A are without $\vee, \exists$. From $\Gamma^g \vdash A^g$ we obtain $\Gamma \vdash_c A$ as follows. We argue informally. Assume Γ. Then Γ^g by the note, hence A^g because of $\Gamma^g \vdash A^g$, hence A again by the note. □

1.2. Normalization

A derivation in normal form does not make "detours", or more precisely, it cannot occur that an elimination rule immediately follows an introduction rule. We will use "conversions" to remove such "local maxima" of complexity, thus reducing any given derivation to normal form. However, there is a

difficulty when we consider an elimination rule for $\vee$, $\wedge$ or $\exists$. An introduced formula may be used as a minor premise of an application of $\vee^-$, $\wedge^-$ or $\exists^-$, then stay the same throughout a sequence of applications of these rules, being eliminated at the end. This also constitutes a local maximum, which we should like to eliminate; *permutative conversions* are designed for exactly this situation. In a permutative conversion we permute an E-rule upwards over the minor premises of $\vee^-$, $\wedge^-$ or $\exists^-$.

We analyse the shape of derivations in normal form, and then prove the (crucial) subformula property, which says that every formula in a normal derivation is a subformula of the end-formula or else of an assumption.

It will be convenient to represent derivations as typed "derivation terms", where the derived formula is seen as the "type" of the term (and displayed as a superscript). This representation is known under the name *Curry-Howard correspondence*. We give an inductive definition of such derivation terms for the $\rightarrow, \forall$-rules in table 1 where for clarity we have written the corresponding derivations to the left. In table 2 this is extended to the rules for $\vee$, $\wedge$ and $\exists$.

1.2.1. Conversions. A conversion eliminates a detour in a derivation, i.e., an elimination immediately following an introduction. We now spell out in detail which conversions we shall allow. This is done for derivations written in tree notation and also as derivation terms.

$\rightarrow$*-conversion.*

$$\dfrac{\dfrac{\begin{array}{c}[u\colon A]\\ \vert\, M\\ B\end{array}}{A \rightarrow B}\rightarrow^+ u \qquad \begin{array}{c}\vert\, N\\ A\end{array}}{B}\rightarrow^- \quad \mapsto \quad \begin{array}{c}\vert\, N\\ A\\ \vert\, M\\ B\end{array}$$

or written as derivation terms $(\lambda_u M(u^A)^B)^{A\rightarrow B} N^A \mapsto M(N^A)^B$. The reader familiar with λ-calculus should note that this is nothing other than β-conversion.

$\forall$*-conversion.*

$$\dfrac{\dfrac{\begin{array}{c}\vert\, M\\ A(x)\end{array}}{\forall_x A(x)}\forall^+ x \qquad r}{A(r)}\forall^- \quad \mapsto \quad \begin{array}{c}\vert\, M'\\ A(r)\end{array}$$

or written as derivation terms $(\lambda_x M(x)^{A(x)})^{\forall_x A(x)} r \mapsto M(r)$.

Derivation	Term
$u\colon A$	u^A
$\dfrac{\begin{array}{c}[u\colon A]\\ \mid M\\ B\end{array}}{A \to B} \to^+ u$	$(\lambda_{u^A} M^B)^{A\to B}$
$\dfrac{\begin{array}{cc}\mid M & \mid N\\ A \to B & A\end{array}}{B} \to^-$	$(M^{A\to B} N^A)^B$
$\dfrac{\begin{array}{c}\mid M\\ A\end{array}}{\forall_x A} \forall^+ x$ (with var.cond.)	$(\lambda_x M^A)^{\forall_x A}$ (with var.cond.)
$\dfrac{\begin{array}{cc}\mid M & \\ \forall_x A(x) & r\end{array}}{A(r)} \forall^-$	$(M^{\forall_x A(x)} r)^{A(r)}$

TABLE 1. Derivation terms for $\to$ and $\forall$

$\vee$-*conversion.*

$$\dfrac{\dfrac{\begin{array}{c}\mid M\\ A\end{array}}{A \vee B} \vee_0^+ \qquad \begin{array}{c}[u\colon A]\\ \mid N\\ C\end{array} \qquad \begin{array}{c}[v\colon B]\\ \mid K\\ C\end{array}}{C} \vee^- u, v \quad \mapsto \quad \begin{array}{c}\mid M\\ A\\ \mid N\\ C\end{array}$$

or as derivation terms $(\vee_{0,B}^+ M^A)^{A\vee B}(u^A.N(u)^C, v^B.K(v)^C) \mapsto N(M^A)^C$, and similarly for $\vee_1^+$ with K instead of N.

Derivation	Term
$\dfrac{\begin{array}{c}\mid M\\ A\end{array}}{A \vee B}\vee_0^+ \qquad \dfrac{\begin{array}{c}\mid M\\ B\end{array}}{A \vee B}\vee_1^+$	$(\vee_{0,B}^+ M^A)^{A\vee B} \quad (\vee_{1,A}^+ M^B)^{A\vee B}$
$\dfrac{\begin{array}{ccc}\mid M & [u\colon A] & [v\colon B]\\ A \vee B & \mid N & \mid K\\ & C & C\end{array}}{C}\vee^- u, v$	$(M^{A\vee B}(u^A.N^C, v^B.K^C))^C$
$\dfrac{\begin{array}{cc}\mid M & \mid N\\ A & B\end{array}}{A \wedge B}\wedge^+$	$\langle M^A, N^B\rangle^{A\wedge B}$
$\dfrac{\begin{array}{ccc}\mid M & [u\colon A] & [v\colon B]\\ A \wedge B & & \mid N\\ & & C\end{array}}{C}\wedge^- u, v$	$(M^{A\wedge B}(u^A, v^B.N^C))^C$
$\dfrac{\begin{array}{cc}\mid M\\ r & A(r)\end{array}}{\exists_x A(x)}\exists^+$	$(\exists_{x,A}^+ r M^{A(r)})^{\exists_x A(x)}$
$\dfrac{\begin{array}{cc}[u\colon A]\\ \mid M & \mid N\\ \exists_x A & B\end{array}}{B}\exists^- x, u$ (var.cond.)	$(M^{\exists_x A}(u^A.N^B))^B$ (var.cond.)

TABLE 2. Derivation terms for $\vee$, $\wedge$ and $\exists$

$\wedge$-*conversion.*

$$\dfrac{\dfrac{\begin{array}{c}|\,M\\A\end{array}\qquad\begin{array}{c}|\,N\\B\end{array}}{A\wedge B}\wedge^{+}\qquad\begin{array}{c}[u\colon A]\quad[v\colon B]\\|\,K\\C\end{array}}{C}\wedge^{-}u,v \quad\mapsto\quad \begin{array}{c}|\,M\\A\end{array}\qquad\begin{array}{c}|\,N\\B\end{array}\qquad\begin{array}{c}|\,K\\C\end{array}$$

or $\langle M^A, N^B\rangle^{A\wedge B}(u^A, v^B.K(u,v)^C) \mapsto K(M^A, N^B)^C$.

$\exists$-*conversion.*

$$\dfrac{\dfrac{r\qquad\begin{array}{c}|\,M\\A(r)\end{array}}{\exists_x A(x)}\exists^{+}\qquad\begin{array}{c}[u\colon A(x)]\\|\,N\\B\end{array}}{B}\exists^{-}x,u \quad\mapsto\quad \begin{array}{c}|\,M\\A(r)\\|\,N'\\B\end{array}$$

or $(\exists^{+}_{x,A} r M^{A(r)})^{\exists_x A(x)}(u^{A(x)}.N(x,u)^B) \mapsto N(r, M^{A(r)})^B$.

1.2.2. Permutative conversions.

$\vee$-*permutative conversion.*

$$\dfrac{\begin{array}{c}|\,M\\A\vee B\end{array}\qquad\dfrac{\begin{array}{c}|\,N\\C\end{array}\qquad\begin{array}{c}|\,K\\C\end{array}}{C}\qquad\begin{array}{c}|\,L\\C'\end{array}}{D}\text{ E-rule} \quad\mapsto$$

$$\dfrac{\begin{array}{c}|\,M\\A\vee B\end{array}\qquad\dfrac{\begin{array}{c}|\,N\\C\end{array}\qquad\begin{array}{c}|\,L\\C'\end{array}}{D}\text{ E-rule}\qquad\dfrac{\begin{array}{c}|\,K\\C\end{array}\qquad\begin{array}{c}|\,L\\C'\end{array}}{D}\text{ E-rule}}{D}$$

or with for instance $\to^{-}$ as E-rule $(M^{A\vee B}(u^A.N^{C\to D}, v^B.K^{C\to D}))^{C\to D}L^C \mapsto (M^{A\vee B}(u^A.(N^{C\to D}L^C)^D, v^B.(K^{C\to D}L^C)^D))^D$.

$\wedge$-*permutative conversion.*

$$\dfrac{\dfrac{\begin{array}{c}|\,M\\A\wedge B\end{array}\qquad\begin{array}{c}|\,N\\C\end{array}}{C}\qquad\begin{array}{c}|\,K\\C'\end{array}}{D}\text{ E-rule} \quad\mapsto$$

$$\dfrac{\begin{array}{c}|\,M\\A\wedge B\end{array}\qquad\dfrac{\begin{array}{c}|\,N\\C\end{array}\qquad\begin{array}{c}|\,K\\C'\end{array}}{D}\text{ E-rule}}{D}$$

or $(M^{A\wedge B}(u^A, v^B.N^{C\to D}))^{C\to D}K^C \mapsto (M^{A\wedge B}(u^A, v^B.(N^{C\to D}K^C)^D))^D$.

$\exists$-*permutative conversion.*

$$\dfrac{\dfrac{\begin{array}{c}\mid M\\ \exists_x A\end{array}\qquad\begin{array}{c}\mid N\\ B\end{array}}{B}\qquad\begin{array}{c}\mid K\\ C\end{array}}{D}\,\text{E-rule}\quad\mapsto$$

$$\dfrac{\begin{array}{c}\mid M\\ \exists_x A\end{array}\qquad\dfrac{\begin{array}{c}\mid N\\ B\end{array}\qquad\begin{array}{c}\mid K\\ C\end{array}}{D}\,\text{E-rule}}{D}$$

or $(M^{\exists_x A}(u^A.N^{C\to D}))^{C\to D}K^C \mapsto (M^{\exists_x A}(u^A.(N^{C\to D}K^C)^D))^D$.

1.2.3. Simplification conversions. These are somewhat trivial conversions, which remove unnecessary applications of the elimination rules for $\vee$, $\wedge$ and $\exists$. For $\vee$ we have

$$\dfrac{\begin{array}{c}\mid M\\ A\vee B\end{array}\qquad\begin{array}{c}[u\colon A]\\ \mid N\\ C\end{array}\qquad\begin{array}{c}[v\colon B]\\ \mid K\\ C\end{array}}{C}\vee^- u, v\quad\mapsto\quad\begin{array}{c}\mid N\\ C\end{array}$$

if $u\colon A$ is not free in N, or $(M^{A\vee B}(u^A.N^C, v^B.K^C))^C \mapsto N^C$; similar for the second component. For $\wedge$ there is the conversion

$$\dfrac{\begin{array}{c}\mid M\\ A\wedge B\end{array}\qquad\begin{array}{c}[u\colon A]\\ [v\colon B]\\ \mid N\\ C\end{array}}{C}\wedge^- u, v\quad\mapsto\quad\begin{array}{c}\mid N\\ C\end{array}$$

if neither $u\colon A$ nor $v\colon B$ is free in N, or $(M^{A\wedge B}(u^A, v^B.N^C))^C \mapsto N^C$. For $\exists$ the simplification conversion is

$$\dfrac{\begin{array}{c}\mid M\\ \exists_x A\end{array}\qquad\begin{array}{c}[u\colon A]\\ \mid N\\ B\end{array}}{B}\exists^- x, u\quad\mapsto\quad\begin{array}{c}\mid N\\ B\end{array}$$

if again $u\colon A$ is not free in N, or $(M^{\exists_x A}(u^A.N^B))^B \mapsto N^B$.

1.2.4. Strong normalization. We now show that no matter in which order we apply the conversion rules, they will always terminate and produce a derivation in "normal form", where no further conversions can be applied.

We shall write derivation terms without formula super- or subscripts. For instance, we write $\exists^+$ instead of $\exists^+_{x,A}$. Hence we consider derivation

terms M, N, K now of the forms

$$u \mid \lambda_v M \mid \lambda_y M \mid \vee_0^+ M \mid \vee_1^+ M \mid \langle M, N\rangle \mid \exists^+ rM \mid$$
$$MN \mid Mr \mid M(v_0.N_0, v_1.N_1) \mid M(v, w.N) \mid M(v.N)$$

where, in these expressions, the variables v, y, v_0, v_1, w are bound.

To simplify the technicalities, we restrict our treatment to the rules for $\to$ and $\exists$. The argument easily extends to the full set of rules. Hence we consider

$$u \mid \lambda_v M \mid \exists^+ rM \mid MN \mid M(v.N).$$

The strategy for strong normalization is set out below, but a word about notation is crucial here. Whenever we write an applicative term as $M\vec{N} := MN_1 \dots N_k$ the convention is that bracketing to the left operates. That is, $M\vec{N} = (\dots(MN_1)\dots N_k)$.

We reserve the letters E, F, G for *eliminations*, i.e., expressions of the form $(v.N)$, and R, S, T for both terms and eliminations. Using this notation we obtain a second (and clearly equivalent) inductive definition of terms:

$$u\vec{M} \mid u\vec{M}E \mid \lambda_v M \mid \exists^+ rM \mid$$
$$(\lambda_v M)N\vec{R} \mid \exists^+ rM(v.N)\vec{R} \mid u\vec{M}ER\vec{S}.$$

Here only the final three forms are not normal: $(\lambda_v M)N\vec{R}$ and $\exists^+ rM(v.N)\vec{R}$ both are *β-redexes*, and $u\vec{M}ER\vec{S}$ is a *permutative redex*. The conversion rules for them are

$(\lambda_v M(v))N$	$\mapsto_\beta M(N)$	$\beta_\to$-conversion,
$\exists^+_{x,A} rM(v.N(x,v))$	$\mapsto_\beta N(r, M)$	$\beta_\exists$-conversion,
$M(v.N)R$	$\mapsto_\pi M(v.NR)$	permutative conversion.

In addition we also allow

$$M(v.N) \mapsto_\sigma N \quad \text{if } v\colon A \text{ is not free in } N.$$

The latter is called a *simplification conversion*, and $M(v.N)$ a *simplification redex*.

The *closure* of these conversions is defined by

(a) If $M \mapsto_\xi M'$ for $\xi = \beta, \pi, \sigma$, then $M \to M'$.
(b) If $M \to M'$, then $MR \to M'R$, $NM \to NM'$, $N(v.M) \to N(v.M')$, $\lambda_v M \to \lambda_v M'$, $\exists^+ rM \to \exists^+ rM'$ (*inner reductions*).

So $M \to N$ means that M *reduces in one step to* N, i.e., N is obtained from M by replacement of (an occurrence of) a redex M' of M by a conversum M'' of M', i.e., by a single conversion. The relation $\to^+$ ("*properly reduces to*") is the transitive closure of $\to$, and $\to^*$ ("*reduces to*") is the reflexive transitive closure of $\to$. A term M is *in normal form* (or simply *normal*) if

M does not contain a redex. *M has a normal form* if there is a normal N such that $M \to^* N$. A *reduction sequence* is a (finite or infinite) sequence $M_0, M_1, M_2 \ldots$ such that $M_i \to M_{i+1}$, for all i.

We inductively define a set SN of derivation terms. In doing so we take care that for a given M there is exactly one rule applicable to generate $M \in \mathrm{SN}$. This will be crucial to make the later proofs work.

Definition (SN).

$$\frac{\vec{M} \in \mathrm{SN}}{u\vec{M} \in \mathrm{SN}}\,(\mathrm{Var}_0) \quad \frac{M \in \mathrm{SN}}{\lambda_v M \in \mathrm{SN}}\,(\lambda) \quad \frac{M \in \mathrm{SN}}{\exists^+ r M \in \mathrm{SN}}\,(\exists)$$

$$\frac{\vec{M}, N \in \mathrm{SN}}{u\vec{M}(v.N) \in \mathrm{SN}}\,(\mathrm{Var}) \quad \frac{u\vec{M}(v.NR)\vec{S} \in \mathrm{SN}}{u\vec{M}(v.N)R\vec{S} \in \mathrm{SN}}\,(\mathrm{Var}_\pi)$$

$$\frac{M(N)\vec{R} \in \mathrm{SN} \qquad N \in \mathrm{SN}}{(\lambda_v M(v))N\vec{R} \in \mathrm{SN}}\,(\beta_\to)$$

$$\frac{N(r, M)\vec{R} \in \mathrm{SN} \qquad M \in \mathrm{SN}}{\exists^+_{x,A} r M(v.N(x, v))\vec{R} \in \mathrm{SN}}\,(\beta_\exists)$$

In (Var_π) we require that x (from $\exists_x A$) and v are not free in R.

It is easy to see that SN is closed under substitution for object variables: if $M(x) \in \mathrm{SN}$, then $M(r) \in \mathrm{SN}$. The proof is by induction on $M \in \mathrm{SN}$, applying the induction hypothesis first to the premise(es) and then reapplying the same rule.

We write $M{\downarrow}$ to mean that M is strongly normalizing, i.e., that every reduction sequence starting from M terminates. By analysing the possible reduction steps we now show that the set $\{\, M \mid M{\downarrow} \,\}$ has the closure properties of the definition of SN above, and hence $\mathrm{SN} \subseteq \{\, M \mid M{\downarrow} \,\}$.

Lemma. *Every term in* SN *is strongly normalizing.*

Proof. We distinguish cases according to the generation rule of SN applied last. The following rules deserve special attention.

Case (Var_π). We prove, as an auxiliary lemma, that

$$u\vec{M}(v.NR)\vec{S}{\downarrow} \text{ implies } u\vec{M}(v.N)R\vec{S}{\downarrow}.$$

As a typical case consider

$$u\vec{M}(v.N(v'.N'))TS{\downarrow} \text{ implies } u\vec{M}(v.N)(v'.N')TS{\downarrow}.$$

However, it is easy to see that any infinite reduction sequence of the latter would give rise to an infinite reduction sequence of the former.

Case ($\beta_\to$). We show that $M(N)\vec{R}\downarrow$ and $N\downarrow$ imply $(\lambda_v M(v))N\vec{R}\downarrow$. This is done by induction on $N\downarrow$, with a side induction on $M(N)\vec{R}\downarrow$. We need to consider all possible reducts of $(\lambda_v M(v))N\vec{R}$. In case of an outer β-reduction use the assumption. If N is reduced, use the induction hypothesis. Reductions in M and in $\vec{R}$ as well as permutative reductions within $\vec{R}$ are taken care of by the side induction hypothesis.

Case ($\beta_\exists$). We show that

$$N(r,M)\vec{R}\downarrow \text{ and } M\downarrow \text{ together imply } \exists^+ rM(v.N(x,v))\vec{R}\downarrow.$$

This is done by a threefold induction: first on $M\downarrow$, second on $N(r,M)\vec{R}\downarrow$ and third on the length of $\vec{R}$. We need to consider all possible reducts of $\exists^+ rM(v.N(x,v))\vec{R}$. In case of an outer β-reduction it must reduce to $N(r,M)\vec{R}$, hence the result by assumption. If M is reduced, use the first induction hypothesis. Reductions in $N(x,v)$ and in $\vec{R}$ as well as permutative reductions within $\vec{R}$ are taken care of by the second induction hypothesis. The only remaining case is when $\vec{R} = S\vec{S}$ and $(v.N(x,v))$ is permuted with S, to yield $\exists^+ rM(v.N(x,v)S)\vec{S}$, in which case the third induction hypothesis applies. □

For later use we prove a slightly generalized form of the rule (Var_π):

PROPOSITION. *If* $M(v.NR)\vec{S} \in \mathrm{SN}$, *then* $M(v.N)R\vec{S} \in \mathrm{SN}$.

PROOF. Induction on the generation of $M(v.NR)\vec{S} \in \mathrm{SN}$. We distinguish cases according to the form of M.

Case $u\vec{T}(v.NR)\vec{S} \in \mathrm{SN}$. If $\vec{T} = \vec{M}$ (i.e., $\vec{T}$ consists of derivation terms only), use (Var_π). Else we have $u\vec{M}(v'.N')\vec{R}(v.NR)\vec{S} \in \mathrm{SN}$. This must be generated by repeated applications of (Var_π) from $u\vec{M}(v'.N'\vec{R}(v.NR)\vec{S}) \in \mathrm{SN}$, and finally by (Var) from $\vec{M} \in \mathrm{SN}$ and $N'\vec{R}(v.NR)\vec{S} \in \mathrm{SN}$. The induction hypothesis for the latter fact yields $N'\vec{R}(v.N)R\vec{S} \in \mathrm{SN}$, hence $u\vec{M}(v'.N'\vec{R}(v.N)R\vec{S}) \in \mathrm{SN}$ by (Var) and finally $u\vec{M}(v'.N')\vec{R}(v.N)R\vec{S} \in \mathrm{SN}$ by (Var_π).

Case $\exists^+ rM\vec{T}(v.N(x,v)R)\vec{S} \in \mathrm{SN}$. Similar, with ($\beta_\exists$) instead of ($\mathrm{Var}_\pi$). In detail: If $\vec{T}$ is empty, by ($\beta_\exists$) this came from $N(r,M)R\vec{S} \in \mathrm{SN}$ and $M \in \mathrm{SN}$, hence $\exists^+ rM(v.N(x,v))R\vec{S} \in \mathrm{SN}$ again by ($\beta_\exists$). Otherwise we have $\exists^+ rM(v'.N'(x',v'))\vec{T}(v.NR)\vec{S} \in \mathrm{SN}$. This must be generated by ($\beta_\exists$) from $N'(r,M)\vec{T}(v.NR)\vec{S} \in \mathrm{SN}$. The induction hypothesis yields $N'(r,M)\vec{T}(v.N)R\vec{S} \in \mathrm{SN}$, hence $\exists^+ rM(v'.N'(x,v'))\vec{T}(v.N)R\vec{S} \in \mathrm{SN}$ by ($\beta_\exists$).

Case $(\lambda_v M(v))N'\vec{R}(w.NR)\vec{S} \in \mathrm{SN}$. By $(\beta_\to)$ this came from $N' \in \mathrm{SN}$ and $M(N')\vec{R}(w.NR)\vec{S} \in \mathrm{SN}$. But the induction hypothesis yields $M(N')\vec{R}(w.N)R\vec{S} \in \mathrm{SN}$, hence $(\lambda_v M(v))N'\vec{R}(w.N)R\vec{S} \in \mathrm{SN}$ by $(\beta_\to)$. □

We show, finally, that *every* term is in SN and hence is strongly normalizing. Given the definition of SN we only have to show that SN is closed under $\to^-$ and $\exists^-$. But in order to prove this we must prove simultaneously the closure of SN under substitution.

THEOREM (Properties of SN). *For all formulas A,*

(a) *for all* $M \in \mathrm{SN}$, *if* M *proves* $A = A_0 \to A_1$ *and* $N \in \mathrm{SN}$, *then* $MN \in \mathrm{SN}$,
(b) *for all* $M \in \mathrm{SN}$, *if* M *proves* $A = \exists_x B$ *and* $N \in \mathrm{SN}$, *then* $M(v.N) \in \mathrm{SN}$,
(c) *for all* $M(v) \in \mathrm{SN}$, *if* $N^A \in \mathrm{SN}$, *then* $M(N) \in \mathrm{SN}$.

PROOF. Induction on $|A|$. We prove (a) and (b) before (c), and hence have (a) and (b) available for the proof of (c). More formally, by induction on A we simultaneously prove that (a) holds, that (b) holds and that (a), (b) together imply (c).

(a). By side induction on $M \in \mathrm{SN}$. Let $M \in \mathrm{SN}$ and assume that M proves $A = A_0 \to A_1$ and $N \in \mathrm{SN}$. We distinguish cases according to how $M \in \mathrm{SN}$ was generated. For (Var_0), (Var_π), $(\beta_\to)$ and $(\beta_\exists)$ use the same rule again.

Case $u\vec{M}(v.N') \in \mathrm{SN}$ by (Var) from $\vec{M}, N' \in \mathrm{SN}$. Then $N'N \in \mathrm{SN}$ by side induction hypothesis for N', hence $u\vec{M}(v.N'N) \in \mathrm{SN}$ by (Var), hence $u\vec{M}(v.N')N \in \mathrm{SN}$ by (Var_π).

Case $(\lambda_v M(v))^{A_0 \to A_1} \in \mathrm{SN}$ by (λ) from $M(v) \in \mathrm{SN}$. Use $(\beta_\to)$; for this we need to know $M(N) \in \mathrm{SN}$. But this follows from induction hypothesis (c) for $M(v)$, since N derives A_0.

(b). By side induction on $M \in \mathrm{SN}$. Let $M \in \mathrm{SN}$ and assume that M proves $A = \exists_x B$ and $N \in \mathrm{SN}$. The goal is $M(v.N) \in \mathrm{SN}$. We distinguish cases according to how $M \in \mathrm{SN}$ was generated. For (Var_π), $(\beta_\to)$ and $(\beta_\exists)$ use the same rule again.

Case $u\vec{M} \in \mathrm{SN}$ by (Var_0) from $\vec{M} \in \mathrm{SN}$. Use (Var).

Case $(\exists^+ rM)^{\exists_x A} \in \mathrm{SN}$ by $(\exists)$ from $M \in \mathrm{SN}$. We must show that $\exists^+ rM(v.N(x,v)) \in \mathrm{SN}$. Use $(\beta_\exists)$; for this we need to know $N(r,M) \in \mathrm{SN}$. But this follows from induction hypothesis (c) for $N(r,v)$ (which is in SN by the remark above), since M derives $A(r)$.

Case $u\vec{M}(v'.N') \in \mathrm{SN}$ by (Var) from $\vec{M}, N' \in \mathrm{SN}$. Then $N'(v.N) \in \mathrm{SN}$ by side induction hypothesis for N', hence $u\vec{M}(v.N'(v.N)) \in \mathrm{SN}$ by (Var) and therefore $u\vec{M}(v.N')(v.N) \in \mathrm{SN}$ by (Var_π).

(c). By side induction on $M(v) \in \mathrm{SN}$. Let $N^A \in \mathrm{SN}$; the goal is $M(N) \in \mathrm{SN}$. We distinguish cases according to how $M(v) \in \mathrm{SN}$ was generated. For

(λ), ($\exists$), ($\beta_\to$) and ($\beta_\exists$) use the same rule again, after applying the induction hypothesis to the premise(es).

Case $u\vec{M}(v) \in \mathrm{SN}$ by (Var_0) from $\vec{M}(v) \in \mathrm{SN}$. Then $\vec{M}(N) \in \mathrm{SN}$ by side induction hypothesis (c). If $u \neq v$, use (Var_0) again. If $u = v$, we must show $N\vec{M}(N) \in \mathrm{SN}$. Note that N proves A; hence the claim follows from $\vec{M}(N) \in \mathrm{SN}$ by (a) with $M = N$.

Case $u\vec{M}(v)(v'.N'(v)) \in \mathrm{SN}$ by (Var) from $\vec{M}(v), N'(v) \in \mathrm{SN}$. If $u \neq v$, use (Var) again. If $u = v$, we must show $N\vec{M}(N)(v'.N'(N)) \in \mathrm{SN}$. Note that N proves A; hence in case $\vec{M}(v)$ is empty the claim follows from (b) with $M = N$, and otherwise from (a), (b) and the induction hypothesis.

Case $u\vec{M}(v)(v'.N'(v))R(v)\vec{S}(v) \in \mathrm{SN}$ has been obtained by (Var_π) from $u\vec{M}(v)(v'.N'(v)R(v))\vec{S}(v) \in \mathrm{SN}$. If $u \neq v$, use (Var_π) again. If $u = v$, from the side induction hypothesis we obtain $N\vec{M}(N)(v'.N'(N)R(N))\vec{S}(N) \in \mathrm{SN}$. Now use the proposition above with $M := N\vec{M}(N)$. □

Corollary. *Every derivation term is in* SN *and therefore strongly normalizing.*

Proof. Induction on the (first) inductive definition of derivation terms. In cases u, $\lambda_v M$ and $\exists^+ r M$ the claim follows from the definition of SN, and in cases MN and $M(v.N)$ from parts (a), (b) of the previous theorem. □

1.2.5. On disjunction. Incorporating the full set of rules adds no other technical complications but merely increases the length. For the energetic reader, however, we include here the details necessary for disjunction. The conjunction case is entirely straightforward.

We have additional β-conversions

$$\vee_i^+ M(v_0.N_0, v_1.N_1) \mapsto_\beta N_i[v_i := M] \quad \beta_{\vee_i}\text{-conversion.}$$

The definition of SN needs to be extended by

$$\frac{M \in \mathrm{SN}}{\vee_i^+ M \in \mathrm{SN}}\,(\vee_i)$$

$$\frac{\vec{M}, N_0, N_1 \in \mathrm{SN}}{u\vec{M}(v_0.N_0, v_1.N_1) \in \mathrm{SN}}\,(\mathrm{Var}_\vee) \qquad \frac{u\vec{M}(v_0.N_0R, v_1.N_1R)\vec{S} \in \mathrm{SN}}{u\vec{M}(v_0.N_0, v_1.N_1)R\vec{S} \in \mathrm{SN}}\,(\mathrm{Var}_{\vee,\pi})$$

$$\frac{N_i[v_i := M]\vec{R} \in \mathrm{SN} \qquad N_{1-i}\vec{R} \in \mathrm{SN} \qquad M \in \mathrm{SN}}{\vee_i^+ M(v_0.N_0, v_1.N_1)\vec{R} \in \mathrm{SN}}\,(\beta_{\vee_i})$$

The former rules (Var), (Var_π) should then be renamed into ($\mathrm{Var}_\exists$), ($\mathrm{Var}_{\exists,\pi}$).

The lemma above stating that every term in SN is strongly normalizable needs to be extended by an additional clause:

Case $(\beta_{\vee_i})$. We show that $N_i[v_i := M]\vec{R}\downarrow$, $N_{1-i}\vec{R}\downarrow$ and $M\downarrow$ together imply $\vee_i^+ M(v_0.N_0, v_1.N_1)\vec{R}\downarrow$. This is done by a fourfold induction: first on $M\downarrow$, second on $N_i[v_i := M]\vec{R}\downarrow$, $N_{1-i}\vec{R}\downarrow$, third on $N_{1-i}\vec{R}\downarrow$ and fourth on the length of $\vec{R}$. We need to consider all possible reducts of $\vee_i^+ M(v_0.N_0, v_1.N_1)\vec{R}$. In case of an outer β-reduction use the assumption. If M is reduced, use the first induction hypothesis. Reductions in N_i and in $\vec{R}$ as well as permutative reductions within $\vec{R}$ are taken care of by the second induction hypothesis. Reductions in N_{1-i} are taken care of by the third induction hypothesis. The only remaining case is when $\vec{R} = S\vec{S}$ and $(v_0.N_0, v_1.N_1)$ is permuted with S, to yield $(v_0.N_0S, v_1.N_1S)$. Apply the fourth induction hypothesis, since $(N_iS)[v := M]\vec{S} = N_i[v := M]S\vec{S}$.

Finally the theorem above stating properties of SN needs an additional clause:

(b') for all $M \in \mathrm{SN}$, if M proves $A = A_0 \vee A_1$ and $N_0, N_1 \in \mathrm{SN}$, then $M(v_0.N_0, v_1.N_1) \in \mathrm{SN}$.

Proof. The new clause is proved by induction on $M \in \mathrm{SN}$. Let $M \in \mathrm{SN}$ and assume that M proves $A = A_0 \vee A_1$ and $N_0, N_1 \in \mathrm{SN}$. The goal is $M(v_0.N_0, v_1.N_1) \in \mathrm{SN}$. We distinguish cases according to how $M \in \mathrm{SN}$ was generated. For $(\mathrm{Var}_{\exists,\pi})$, $(\mathrm{Var}_{\vee,\pi})$, $(\beta_\to)$, $(\beta_\exists)$ and $(\beta_{\vee_i})$ use the same rule again.

Case $u\vec{M} \in \mathrm{SN}$ by (Var_0) from $\vec{M} \in \mathrm{SN}$. Use $(\mathrm{Var}_\vee)$.

Case $(\vee_i^+ M)^{A_0 \vee A_1} \in \mathrm{SN}$ by $(\vee_i)$ from $M \in \mathrm{SN}$. Use $(\beta_{\vee_i})$; for this we need to know $N_i[v_i := M] \in \mathrm{SN}$ and $N_{1-i} \in \mathrm{SN}$. The latter is assumed, and the former follows from main induction hypothesis (with N_i) for the substitution clause of the theorem, since M derives A_i.

Case $u\vec{M}(v'.N') \in \mathrm{SN}$ by $(\mathrm{Var}_\exists)$ from $\vec{M}, N' \in \mathrm{SN}$. For brevity let $E := (v_0.N_0, v_1.N_1)$. Then $N'E \in \mathrm{SN}$ by side induction hypothesis for N', so $u\vec{M}(v'.N'E) \in \mathrm{SN}$ by $(\mathrm{Var}_\exists)$ and therefore $u\vec{M}(v'.N')E \in \mathrm{SN}$ by $(\mathrm{Var}_{\exists,\pi})$.

Case $u\vec{M}(v_0'.N_0', v_1'.N_1') \in \mathrm{SN}$ by $(\mathrm{Var}_\vee)$ from $\vec{M}, N_0', N_1' \in \mathrm{SN}$. Let $E := (v_0.N_0, v_1.N_1)$. Then $N_i'E \in \mathrm{SN}$ by side induction hypothesis for N_i', so $u\vec{M}(v_0'.N_0'E, v_1'.N_1'E) \in \mathrm{SN}$ by $(\mathrm{Var}_\vee)$ and therefore $u\vec{M}(v_0'.N_0', v_1'.N_1')E \in \mathrm{SN}$ by $(\mathrm{Var}_{\vee,\pi})$.

Clause (c) now needs additional cases, e.g.,

Case $u\vec{M}(v_0.N_0, v_1.N_1) \in \mathrm{SN}$ by $(\mathrm{Var}_\vee)$ from $\vec{M}, N_0, N_1 \in \mathrm{SN}$. If $u \neq v$, use $(\mathrm{Var}_\vee)$. If $u = v$, we show $N\vec{M}[v := N](v_0.N_0[v := N], v_1.N_1[v := N]) \in$

SN. Note that N proves A; hence in case $\vec{M}$ empty the claim follows from (b), and otherwise from (a) and the induction hypothesis. □

1.2.6. The structure of normal derivations. To analyse normal derivations, it will be useful to introduce the notions of a *segment* and of a *track* in a proof tree, which make sense for non-normal derivations as well.

DEFINITION. A *segment* (of *length* n) in a derivation M is a sequence $A_0, \ldots, A_n$ of occurrences of the same formula A such that

(a) for $0 \leq i < n$, A_i is a minor premise of an application of $\vee^-$, $\wedge^-$ or $\exists^-$, with conclusion A_{i+1};
(b) A_n is not a minor premise of $\vee^-$, $\wedge^-$ or $\exists^-$.
(c) A_0 is not the conclusion of $\vee^-$, $\wedge^-$ or $\exists^-$.

Notice that a formula occurrence (f.o.) which is neither a minor premise nor the conclusion of an application of $\vee^-$, $\wedge^-$ or $\exists^-$ always constitutes a segment of length 1. A segment is *maximal* or a *cut (segment)* if A_n is the major premise of an E-rule, and either $n > 0$, or $n = 0$ and $A_0 = A_n$ is the conclusion of an I-rule.

We use σ, σ' for segments. σ is called a *subformula* of σ' if the formula A in σ is a subformula of B in σ'.

The notion of a track is designed to retain the subformula property in case one passes through the major premise of an application of a $\vee^-, \wedge^-, \exists^-$-rule. In a track, when arriving at an A_i which is the major premise of an application of such a rule, we take for A_{i+1} a hypothesis discharged by this rule.

DEFINITION. A *track* of a derivation M is a sequence of f.o.'s $A_0, \ldots, A_n$ such that

(a) A_0 is a top f.o. in M not discharged by an application of an $\vee^-, \wedge^-, \exists^-$-rule;
(b) A_i for $i < n$ is not the minor premise of an instance of $\to^-$, and *either*
  (i) A_i is not the major premise of an instance of a $\vee^-, \wedge^-, \exists^-$-rule and A_{i+1} is directly below A_i, *or*
  (ii) A_i is the major premise of an instance of a $\vee^-, \wedge^-, \exists^-$-rule and A_{i+1} is an assumption discharged by this instance;
(c) A_n is *either*
  (i) the minor premise of an instance of $\to^-$, *or*
  (ii) the end formula of M, *or*
  (iii) the major premise of an instance of a $\vee^-, \wedge^-, \exists^-$-rule in case there are no assumptions discharged by this instance.

LEMMA. *In a derivation each formula occurrence belongs to some track.*

PROOF. By induction on derivations. For example, suppose a derivation K ends with an $\exists^-$-application:

$$\dfrac{\exists_x A \qquad \begin{matrix}[u\colon A]\\ \mid N\\ B\end{matrix}}{B}\,\exists^- x, u$$

B in N belongs to a track π (induction hypothesis); either this does not start in $u\colon A$, and then π, B is a track in K which ends in the end formula; or π starts in $u\colon A$, and then there is a track π' in M (induction hypothesis) such that π', π, B is a track in K ending in the end formula. The other cases are left to the reader. □

DEFINITION. A *track of order* 0, or *main track*, in a derivation is a track ending either in the end formula of the whole derivation or in the major premise of an application of a $\vee^-$, $\wedge^-$ or $\exists^-$-rule, provided there are no assumption variables discharged by the application. A *track of order* $n+1$ is a track ending in the minor premise of an $\rightarrow^-$-application, with major premise belonging to a track of order n.

A *main branch* of a derivation is a branch π (i.e., a linearly ordered subtree) in the proof tree such that π passes only through premises of I-rules and *major premises* of E-rules, and π begins at a top node and ends in the end formula.

Since by simplification conversions we have removed every application of an $\vee^-$, $\wedge^-$ or $\exists^-$-rule that discharges no assumption variables, each track of order 0 in a normal derivation is a track ending in the end formula of the whole derivation. Note also that if we search for a main branch going upwards from the end formula, the branch to be followed is unique as long as we do not encounter an $\wedge^+$-application. Now let us consider normal derivations. Recall the notion of a strictly positive part of a formula, defined in 1.1.3.

PROPOSITION. *Let M be a normal derivation, and let $\pi = \sigma_0, \ldots, \sigma_n$ be a track in M. Then there is a segment σ_i in π, the* minimum *segment or* minimum *part of the track, which separates two (possibly empty) parts of π, called the E-part (*elimination part*) and the I-part (*introduction part*) of π such that*

(a) *for each σ_j in the E-part one has $j < i$, σ_j is a major premise of an E-rule, and σ_{j+1} is a strictly positive part of σ_j, and therefore each σ_j is a s.p.p. of σ_0;*

(b) *for each σ_j which is in the I-part or is the minimum segment one has $i \leq j$, and if $j \neq n$, then σ_j is a premise of an I-rule and a s.p.p. of σ_{j+1}, so each σ_j is a s.p.p. of σ_n.*

PROOF. By tracing through the definitions. □

THEOREM (Subformula property). *Let M be a normal derivation. Then each formula occurring in the derivation is a subformula of either the end formula or else an (uncancelled) assumption formula.*

PROOF. As noted above, each track of order 0 in M is a track ending in the end formula of M. Furthermore each track has an E-part above an I-part. Therefore any formula on a track of order 0 is either a subformula of the end formula or else a subformula of an (uncancelled) assumption. We can now prove the theorem for tracks of order n, by induction on n. So assume the result holds for tracks of order n. If A is any formula on a track of order $n+1$, either A lies in the E-part in which case it is a subformula of an assumption, or else it lies in the I-part and is therefore a subformula of the minor premise of an $\rightarrow^-$ whose main premise belongs to a track of order n. In this case A is a subformula of a formula on a track of order n and we can apply the induction hypothesis. □

THEOREM (Disjunction property). *If no strictly positive part of a formula in Γ is a disjunction, then $\Gamma \vdash A \vee B$ implies $\Gamma \vdash A$ or $\Gamma \vdash B$.*

PROOF. Consider a normal derivation M of $A \vee B$ from assumptions Γ not containing a disjunction as s.p.p. The end formula $A \vee B$ is the final formula of a (main) track. If the I-part of this track is empty, then the structure of main tracks ensures that $A \vee B$ would be a s.p.p. of an assumption in Γ, but this is not allowed. Hence $A \vee B$ lies in the I-part of a main track. If above $A \vee B$ this track goes through a minor premise of an $\vee^-$, then the major premise would again be a disjunctive s.p.p. of an assumption, which is not allowed. Thus $A \vee B$ belongs to a segment within the I-part of the track, above which there can only be finitely many $\exists^-$ and $\wedge^-$ followed by an $\vee_i^+$. Its premise is either A or B, and therefore we can replace the segment of $A \vee B$'s by a segment of A's or a segment of B's, thus transforming the proof into either a proof of A or a proof of B. □

There is a similar theorem for the existential quantifier:

THEOREM (Explicit definability under hypotheses). *If no strictly positive part of a formula in Γ is existential, then $\Gamma \vdash \exists_x A(x)$ implies $\Gamma \vdash A(r_1) \vee \cdots \vee A(r_n)$ for some terms $r_1, \ldots, r_n$. If in addition no s.p.p. of a formula in Γ is disjunctive then $\Gamma \vdash \exists_x A(x)$ implies there is even a single term r such that $\Gamma \vdash A(r)$.*

PROOF. Consider a normal derivation M of $\exists_x A(x)$ from assumptions Γ not containing an existential s.p.p. We use induction on the derivation, and distinguish cases on the last rule.

By assumption the last rule cannot be $\exists^-$, using a similar argument to the above. Again as before, the only critical case is when the last rule is $\vee^-$.

$$\dfrac{\begin{array}{ccc} & [u\colon B] & [v\colon C] \\ \mid M & \mid N_0 & \mid N_1 \\ B \vee C & \exists_x A(x) & \exists_x A(x) \end{array}}{\exists_x A(x)} \vee^- u, v$$

By assumption again neither B nor C can have an existential s.p.p. Applying the induction hypothesis to N_0 and N_1 we obtain

$$\dfrac{\begin{array}{ccc} & [u\colon B] & [v\colon C] \\ & \mid & \mid \\ \mid M & \dfrac{\bigvee_{i=1}^{n} A(r_i)}{\bigvee_{i=1}^{n+m} A(r_i)} \vee^+ & \dfrac{\bigvee_{i=n+1}^{n+m} A(r_i)}{\bigvee_{i=1}^{n+m} A(r_i)} \vee^+ \\ B \vee C & & \end{array}}{\bigvee_{i=1}^{n+m} A(r_i)} \vee^- u, v$$

The remaining cases are left to the reader.

The second part of the theorem is proved similarly; by assumption the last rule can be neither $\vee^-$ nor $\exists^-$, so it may be an $\wedge^-$. In that case there is only one minor premise and so no need to duplicate instances of $A(x)$. □

1.3. Soundness and Completeness for Tree Models

It is an obvious question to ask whether the logical rules we have been considering suffice, i.e., whether we have forgotten some necessary rules. To answer this question we first have to fix the *meaning* of a formula, i.e., provide a semantics. This will be done by means of the tree models introduced by Beth (1956). Using this concept of a model we will prove soundness and completeness.

1.3.1. Tree models. Consider a finitely branching tree of "possible worlds". The worlds are represented as nodes in this tree. They may be thought of as possible states such that all nodes "above" a node k are the ways in which k may develop in the future. The worlds are increasing, that is, if an atomic formula $R\vec{s}$ is true in a world k, then $R\vec{s}$ is true in all future worlds k'.

More formally, each tree model is based on a finitely branching tree T. A *node* k over a set S is a finite sequence $k = \langle a_0, a_1, \ldots, a_{n-1}\rangle$ of elements of S; $\mathrm{lh}(k)$ is the length of k. We write $k \preceq k'$ if k is an initial segment of k'. A

tree on S is a set of nodes closed under initial segments. A tree T is *finitely branching* if every node in T has finitely many immediate successors. A tree T is *infinite* if for every $n \in \mathbb{N}$ there is a node $k \in T$ such that $\mathrm{lh}(k) = n$. A *branch* of a tree T is a linearly ordered subtree of T, and a *leaf* of T is a node without successors in T. A tree T is *complete* is every node in T has an immediate successor, i.e., T has no leaves.

For the proof of the completeness theorem, the completeness tree over $\{0, 1\}$ (whose branches constitute Cantor space) will suffice. The nodes will be all the finite sequences of 0's and 1's, and the ordering is as above. The root is the empty sequence and $k0$ is the sequence k with the element 0 added at the end; similarly for $k1$.

For the rest of this section, fix a countable formal language $\mathcal{L}$.

Definition. Let T be a finitely branching tree. A *tree model* on T is a triple $\mathcal{T} = (D, I_0, I_1)$ such that

(a) D is a nonempty set;
(b) for every n-ary function symbol f (in the underlying language $\mathcal{L}$), I_0 assigns to f a map $I_0(f)\colon D^n \to D$;
(c) for every n-ary relation symbol R and every node $k \in T$, $I_1(R, k) \subseteq D^n$ is assigned in such a way that monotonicity is preserved:

$$k \preceq k' \to I_1(R, k) \subseteq I_1(R, k').$$

If $n = 0$, then $I_1(R, k)$ is either true or false. There is no special requirement set on $I_1(\bot, k)$. (Recall that minimal logic places no particular constraints on falsum $\bot$.) We write $R^{\mathcal{T}}(\vec{a}, k)$ for $\vec{a} \in I_1(R, k)$, and $|\mathcal{T}|$ to denote the domain D.

It is obvious from the definition that any tree T can be extended to a complete tree $\bar{T}$ (i.e., without leaves), in which for every leaf $k \in T$ all sequences $k0$, $k00$, $k000, \ldots$ are added to T. For every node $k0 \ldots 0$, we then add $I_1(R, k0 \ldots 0) := I_1(R, k)$.

An *assignment* (or variable assignment) in D is a map η assigning to every variable $x \in \mathrm{dom}(\eta)$ a value $\eta(x) \in D$. Finite assignments will be written as $[x_1 := a_1, \ldots, x_n := a_n]$ or else as $[a_1/x_1, \ldots, a_n/x_n]$, with distinct $x_1, \ldots, x_n$. If η is an assignment in D and $a \in D$, let η^a_x be the assignment in D mapping x to a and coinciding with η elsewhere:

$$\eta^a_x(y) := \begin{cases} \eta(y) & \text{if } y \neq x, \\ a & \text{if } y = x. \end{cases}$$

Let a tree model $\mathcal{T} = (D, I_0, I_1)$ and an assignment η in D be given. We define a homomorphic extension of η (denoted by η as well) to terms t whose variables lie in $\mathrm{dom}(\eta)$ by

$$\eta(c) := I_0(c),$$

$$\eta(f(t_1,\dots,t_n)) := I_0(f)(\eta(t_1),\dots,\eta(t_n)).$$

Observe that the extension of η depends on $\mathcal{T}$; we often write $t^{\mathcal{T}}[\eta]$ for $\eta(t)$.

DEFINITION. $\mathcal{T}, k \Vdash A[\eta]$ ($\mathcal{T}$ *forces* A at node k for an assignment η) is defined inductively. We write $k \Vdash A[\eta]$ when it is clear from the context what the underlying model $\mathcal{T}$ is, and $\forall_{k'\succeq_n k}A$ for $\forall_{k'\succeq k}(\mathrm{lh}(k') = \mathrm{lh}(k) + n \to A)$.

$$\begin{aligned}
k \Vdash (R\vec{s})[\eta] &:= \exists_n \forall_{k'\succeq_n k} R^{\mathcal{T}}(\vec{s}^{\mathcal{T}}[\eta], k'),\\
k \Vdash (A \vee B)[\eta] &:= \exists_n \forall_{k'\succeq_n k}(k' \Vdash A[\eta] \vee k' \Vdash B[\eta]),\\
k \Vdash (\exists_x A)[\eta] &:= \exists_n \forall_{k'\succeq_n k} \exists_{a\in|\mathcal{T}|}(k' \Vdash A[\eta^a_x]),\\
k \Vdash (A \to B)[\eta] &:= \forall_{k'\succeq k}(k' \Vdash A[\eta] \to k' \Vdash B[\eta]),\\
k \Vdash (A \wedge B)[\eta] &:= k \Vdash A[\eta] \wedge k \Vdash B[\eta],\\
k \Vdash (\forall_x A)[\eta] &:= \forall_{a\in|\mathcal{T}|}(k \Vdash A[\eta^a_x]).
\end{aligned}$$

Thus in the atomic, disjunctive and existential cases, the set of k' whose length is $\mathrm{lh}(k) + n$ acts as a "bar" in the complete tree. Note that the implicational case is treated differently, and refers to the "unbounded future".

In this definition, the logical connectives $\to, \wedge, \vee, \forall, \exists$ on the left hand side are part of the object language, whereas the same connectives on the right hand side are to be *understood* in the usual sense: they belong to the "metalanguage". It should always be clear from the context whether a formula is part of the object or the metalanguage.

1.3.2. Covering lemma. It is easily seen (using the definition and monotonicity) that from $k \Vdash A[\eta]$ and $k \preceq k'$ we can conclude $k' \Vdash A[\eta]$. The converse is true as well:

LEMMA (Covering).

$$\forall_{k'\succeq_n k}(k' \Vdash A[\eta]) \to k \Vdash A[\eta].$$

PROOF. Induction on A. We write $k \Vdash A$ for $k \Vdash A[\eta]$.

Case $R\vec{s}$. Assume

$$\forall_{k'\succeq_n k}(k' \Vdash R\vec{s}),$$

hence by definition

$$\forall_{k'\succeq_n k}\exists_m \forall_{k''\succeq_m k'} R^{\mathcal{T}}(\vec{s}^{\mathcal{T}}[\eta], k'').$$

Since T is a finitely branching tree,

$$\exists_m \forall_{k'\succeq_m k} R^{\mathcal{T}}(\vec{s}^{\mathcal{T}}[\eta], k').$$

Hence $k \Vdash R\vec{s}$.

The cases $A \vee B$ and $\exists_x A$ are handled similarly.

Case $A \to B$. Let $k' \Vdash A \to B$ for all $k' \succeq k$ with $\mathrm{lh}(k') = \mathrm{lh}(k) + n$. We show

$$\forall_{l \succeq k}(l \Vdash A \to l \Vdash B).$$

Let $l \succeq k$ and $l \Vdash A$. We must show $l \Vdash B$. To this end we apply the induction hypothesis to B and $m := \max(\mathrm{lh}(k) + n, \mathrm{lh}(l))$. So assume $l' \succeq l$ and $\mathrm{lh}(l') = m$. It is sufficient to show $l' \Vdash B$. If $\mathrm{lh}(l') = \mathrm{lh}(l)$, then $l' = l$ and we are done. If $\mathrm{lh}(l') = \mathrm{lh}(k) + n > \mathrm{lh}(l)$, then l' is an extension of l as well as of k and has length $\mathrm{lh}(k) + n$, and hence $l' \Vdash A \to B$ by assumption. Moreover, $l' \Vdash A$, since $l' \succeq l$ and $l \Vdash A$. It follows that $l' \Vdash B$.

The cases $A \wedge B$ and $\forall_x A$ are easy. □

1.3.3. Soundness.

LEMMA (Coincidence). *Let $\mathcal{T}$ be a tree model, t a term, A a formula and η, ξ assignments in $|\mathcal{T}|$.*

(a) *If $\eta(x) = \xi(x)$ for all $x \in \mathrm{vars}(t)$, then $\eta(t) = \xi(t)$.*
(b) *If $\eta(x) = \xi(x)$ for all $x \in \mathrm{FV}(A)$, then $\mathcal{T}, k \Vdash A[\eta]$ if and only if $\mathcal{T}, k \Vdash A[\xi]$.*

PROOF. Induction on terms and formulas. □

LEMMA (Substitution). *Let $\mathcal{T}$ be a tree model, $t, r(x)$ terms, $A(x)$ a formula and η an assignment in $|\mathcal{T}|$. Then*

(a) $\eta(r(t)) = \eta_x^{\eta(t)}(r(x))$.
(b) *$\mathcal{T}, k \Vdash A(t)[\eta]$ if and only if $\mathcal{T}, k \Vdash A(x)[\eta_x^{\eta(t)}]$.*

PROOF. Induction on terms and formulas. □

THEOREM (Soundness). *Let $\Gamma \cup \{A\}$ be a set of formulas such that $\Gamma \vdash A$. Then, if $\mathcal{T}$ is a tree model, k any node and η an assignment in $|\mathcal{T}|$, it follows that $\mathcal{T}, k \Vdash \Gamma[\eta]$ implies $\mathcal{T}, k \Vdash A[\eta]$.*

PROOF. Induction on derivations.

We begin with the axiom schemes $\vee_0^+$, $\vee_1^+$, $\vee^-$, $\wedge^+$, $\wedge^-$, $\exists^+$ and $\exists^-$. $k \Vdash C[\eta]$ is abbreviated $k \Vdash C$, when η is known from the context.

Case $\vee_0^+ \colon A \to A \vee B$. We show $k \Vdash A \to A \vee B$. Assume for $k' \succeq k$ that $k' \Vdash A$. Show: $k' \Vdash A \vee B$. This follows from the definition, since $k' \Vdash A$. The case $\vee_1^+ \colon B \to A \vee B$ is symmetric.

Case $\vee^- \colon A \vee B \to (A \to C) \to (B \to C) \to C$. We show that $k \Vdash A \vee B \to (A \to C) \to (B \to C) \to C$. Assume for $k' \succeq k$ that $k' \Vdash A \vee B$, $k' \Vdash A \to C$ and $k' \Vdash B \to C$ (we can safely assume that k' is the same for all three premises.) Show that $k' \Vdash C$. By definition, there is an n s.t. for all $k'' \succeq_n k'$, $k'' \Vdash A$ or $k'' \Vdash B$. In both cases it follows

that $k'' \Vdash C$, since $k' \Vdash A \to C$ and $k' \Vdash B \to C$. By the covering lemma, $k' \Vdash C$.

The cases $\wedge^+$, $\wedge^-$ are easy.

Case $\exists^+$: $A \to \exists_x A$. We show $k \Vdash (A \to \exists_x A)[\eta]$. Assume $k' \succeq k$ and $k' \Vdash A[\eta]$. We show $k' \Vdash (\exists_x A)[\eta]$. Since $\eta = \eta_x^{\eta(x)}$ there is an $a \in |\mathcal{T}|$ (namely $a := \eta(x)$) such that $k' \Vdash A[\eta_x^a]$. Hence, $k' \Vdash (\exists_x A)[\eta]$.

Case $\exists^-$: $\exists_x A \to \forall_x(A \to B) \to B$ and $x \notin \mathrm{FV}(B)$. We show that $k \Vdash (\exists_x A \to \forall_x(A \to B) \to B)[\eta]$. Assume that $k' \succeq k$ and $k' \Vdash (\exists_x A)[\eta]$ and $k' \Vdash \forall_x(A \to B)[\eta]$. We show $k' \Vdash B[\eta]$. By definition, there is an n such that for all $k'' \succeq_n k'$ we have $a \in |\mathcal{T}|$ and $k'' \Vdash A[\eta_x^a]$. From $k' \Vdash \forall_x(A \to B)[\eta]$ it follows that $k'' \Vdash B[\eta_x^a]$, and since $x \notin \mathrm{FV}(B)$, from the coincidence lemma, $k'' \Vdash B[\eta]$. Then, finally, by the covering lemma $k' \Vdash B[\eta]$.

This concludes the treatment of the axioms. We now consider the rules. In case of the assumption rule $u\colon A$ we have $A \in \Gamma$ and the claim is obvious.

Case $\to^+$. Assume $k \Vdash \Gamma$. We show $k \Vdash A \to B$. Assume $k' \succeq k$ and $k' \Vdash A$. Our goal is $k' \Vdash B$. We have $k' \Vdash \Gamma \cup \{A\}$. Thus, $k' \Vdash B$ by induction hypothesis.

Case $\to^-$. Assume $k \Vdash \Gamma$. The induction hypothesis gives us $k \Vdash A \to B$ and $k \Vdash A$. Hence $k \Vdash B$.

Case $\forall^+$. Assume $k \Vdash \Gamma[\eta]$ and $x \notin \mathrm{FV}(\Gamma)$. We show $k \Vdash (\forall_x A)[\eta]$, i.e., $k \Vdash A[\eta_x^a]$ for an arbitrary $a \in |\mathcal{T}|$. We have

$$\begin{array}{ll} k \Vdash \Gamma[\eta_x^a] & \text{by the coincidence lemma, since } x \notin \mathrm{FV}(\Gamma) \\ k \Vdash A[\eta_x^a] & \text{by induction hypothesis.} \end{array}$$

Case $\forall^-$. Let $k \Vdash \Gamma[\eta]$. We show that $k \Vdash A(t)[\eta]$. This follows from

$$\begin{array}{ll} k \Vdash (\forall_x A(x))[\eta] & \text{by induction hypothesis} \\ k \Vdash A(x)[\eta_x^{\eta(t)}] & \text{by definition} \\ k \Vdash A(t)[\eta] & \text{by the substitution lemma.} \end{array}$$

This concludes the proof. □

1.3.4. Counter models. With soundness at hand, it is easy to build counter models proving that certain formulas are underivable in minimal or intuitionistic logic. A *tree model for intuitionistic logic* is a tree model $\mathcal{T} = (D, I_0, I_1)$ in which $I_1(\bot, k)$ is false for all k. This is equivalent to saying that $\bot$ is never forced:

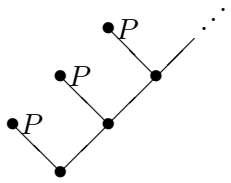
Lemma. *Given any tree model $\mathcal{T}$, $\bot^{\mathcal{T}}(k)$ is false at all nodes k if and only if $k \nVdash \bot$ for all nodes k.*

PROOF. Clearly if $k \Vdash \bot$ then $\bot$ is false at node k. Conversely, suppose $\bot^{\mathcal{T}}(k')$ is false at all nodes k'. We must show $\forall_k(k \Vdash \bot)$. Let k be given. Then, since $\bot^{\mathcal{T}}(k')$ is false at all nodes k', is is certainly false at some $k' \succeq_n k$, for every n. This means $k \Vdash \bot$ by definition. □

Therefore by unravelling the implication clause in the forcing definition, one sees that in any tree model for intuitionistic logic,

$$\begin{aligned}(k \Vdash \neg A) &\leftrightarrow \forall_{k' \succeq k}(k' \nVdash A),\\ (k \Vdash \neg\neg A) &\leftrightarrow \forall_{k' \succeq k}(k' \nVdash \neg A)\\ &\leftrightarrow \forall_{k' \succeq k}\tilde{\exists}_{k'' \succeq k'}(k'' \Vdash A).\end{aligned}$$

As an example we show that $\nvdash_i \neg\neg P \to P$. We describe the desired tree model by means of a diagram below. Next to every node we write all propositions forced at that node.



This is a tree model because monotonicity clearly holds. Observe also that $I_1(\bot, k)$ is false at all nodes k. Hence this is an intuitionistic tree model, and moreover $\langle\rangle \nVdash P$. Using the remark above, it is easily seen that $\langle\rangle \Vdash \neg\neg P$. Thus $\langle\rangle \nVdash (\neg\neg P \to P)$ and hence $\nvdash_i (\neg\neg P \to P)$. The model also shows that the *Peirce formula* $((P \to Q) \to P) \to P$ is not derivable in intuitionistic logic.

As another example we show that the drinker formula $\tilde{\exists}_x(Px \to \forall_x Px)$ from 1.1.8 is intuitionistically underivable, using a quite different tree model. In this case the underlying tree is the full binary one, i.e., its nodes are the finite sequences $k = \langle i_0, i_1, \ldots, i_{n-1}\rangle$ of numbers 0 or 1. For the language determined by $\bot$ and a unary predicate symbol P consider $\mathcal{T} := (D, I_1)$ with $I_1(\bot, k)$ false, $D := \mathbb{N}$ and

$$I_1(P, \langle i_0, \ldots, i_{n-1}\rangle) := \{\, a \in D \mid i_0, \ldots, i_{n-1} \text{ contains at least } a \text{ zeros} \,\}.$$

Cleary $\mathcal{T}$ is an intuitionistic tree model (monotonicity is easily checked), $k \nVdash \forall_x Px$ for every k, and $\forall_{a,k}\exists_{l \succeq k}(l \Vdash Px[x := a])$. Therefore

$$\begin{aligned}&\forall_{a,k}(k \nVdash (Px \to \forall_x Px)[x := a])\\ &\langle\rangle \Vdash \forall_x\neg(Px \to \forall_x Px).\end{aligned}$$

Hence $\nvdash_i \neg\forall_x\neg(Px \to \forall_x Px)$.

1.3.5. Completeness.

THEOREM (Completeness). *Let $\Gamma \cup \{A\}$ be a set of formulas. Then the following propositions are equivalent.*

(a) $\Gamma \vdash A$.

(b) $\Gamma \Vdash A$, *i.e., for all tree models $\mathcal{T}$, nodes k and assignments η*

$$\mathcal{T}, k \Vdash \Gamma[\eta] \to \mathcal{T}, k \Vdash A[\eta].$$

PROOF. Soundness already gives "(a) implies (b)". For the other direction we employ a technique due to Harvey Friedman and construct a tree model $\mathcal{T}$ (over the set T_{01} of all finite 0-1-sequences) whose domain D is the set of all terms of the underlying language, with the property that $\Gamma \vdash B$ is equivalent to $\mathcal{T}, \langle\rangle \Vdash B[\mathrm{id}]$. We can assume here that Γ and also A are closed.

In order to define $\mathcal{T}$, we will need an enumeration $A_0, A_1, A_2, \dots$ of the underlying language $\mathcal{L}$ (assumed countable), in which every formula occurs infinitely often. We also fix an enumeration $x_0, x_1, \dots$ of distinct variables. Since Γ is countable it can we written $\Gamma = \bigcup_n \Gamma_n$ with finite sets Γ_n such that $\Gamma_n \subseteq \Gamma_{n+1}$. With every node $k \in T_{01}$, we associate a finite set Δ_k of formulas and a set V_k of variables, by induction on the length of k.

Let $\Delta_{\langle\rangle} := \emptyset$ and $V_{\langle\rangle} := \emptyset$. Take a node k such that $\mathrm{lh}(k) = n$ and suppose that Δ_k, V_k are already defined. Write $\Delta \vdash_n B$ to mean that there is a derivation of length $\leq n$ of B from Δ. We define Δ_{k0}, V_{k0} and Δ_{k1}, V_{k1} as follows:

Case 0. $\mathrm{FV}(A_n) \not\subseteq V_k$. Then let

$$\Delta_{k0} := \Delta_{k1} := \Delta_k \quad \text{and} \quad V_{k0} := V_{k1} := V_k.$$

Case 1. $\mathrm{FV}(A_n) \subseteq V_k$ and $\Gamma_n, \Delta_k \nvdash_n A_n$. Let

$$\Delta_{k0} := \Delta_k \quad \text{and} \quad \Delta_{k1} := \Delta_k \cup \{A_n\},$$
$$V_{k0} := V_{k1} := V_k.$$

Case 2. $\mathrm{FV}(A_n) \subseteq V_k$ and $\Gamma_n, \Delta_k \vdash_n A_n = A'_n \vee A''_n$. Let

$$\Delta_{k0} := \Delta_k \cup \{A_n, A'_n\} \quad \text{and} \quad \Delta_{k1} := \Delta_k \cup \{A_n, A''_n\},$$
$$V_{k0} := V_{k1} := V_k.$$

Case 3. $\mathrm{FV}(A_n) \subseteq V_k$ and $\Gamma_n, \Delta_k \vdash_n A_n = \exists_x A'_n(x)$. Let

$$\Delta_{k0} := \Delta_{k1} := \Delta_k \cup \{A_n, A'_n(x_i)\} \quad \text{and} \quad V_{k0} := V_{k1} := V_k \cup \{x_i\},$$

where x_i is the first variable $\notin V_k$.

Case 4. $\mathrm{FV}(A_n) \subseteq V_k$ and $\Gamma_n, \Delta_k \vdash_n A_n$, with A_n neither a disjunction nor an existentially quantified formula. Let

$$\Delta_{k0} := \Delta_{k1} := \Delta_k \cup \{A_n\} \quad \text{and} \quad V_{k0} := V_{k1} := V_k.$$

Obviously $\mathrm{FV}(\Delta_k) \subseteq V_k$, and $k \preceq k'$ implies that $\Delta_k \subseteq \Delta_{k'}$. Notice also that because of $\vdash \exists_x(\bot \to \bot)$ and the fact that this formula is repeated infinitely often in the given enumeration, for every variable x_i there is an m such that $x_i \in V_k$ for all k with $\mathrm{lh}(k) = m$.

We note that

$$\forall_{k' \succeq_n k}\,(\Gamma, \Delta_{k'} \vdash B) \to \Gamma, \Delta_k \vdash B, \quad \text{provided } \mathrm{FV}(B) \subseteq V_k. \tag{1.7}$$

It is sufficient to show that, for $\mathrm{FV}(B) \subseteq V_k$,

$$(\Gamma, \Delta_{k0} \vdash B) \wedge (\Gamma, \Delta_{k1} \vdash B) \to (\Gamma, \Delta_k \vdash B).$$

In cases 0, 1 and 4, this is obvious. For case 2, the claim follows immediately from the axiom schema $\vee^-$. In case 3, we have $\mathrm{FV}(A_n) \subseteq V_k$ and $\Gamma_n, \Delta_k \vdash_n A_n = \exists_x A'_n(x)$. Assume $\Gamma, \Delta_k \cup \{A_n, A'_n(x_i)\} \vdash B$ with $x_i \notin V_k$, and $\mathrm{FV}(B) \subseteq V_k$. Then $x_i \notin \mathrm{FV}(\Delta_k \cup \{A_n, B\})$, hence $\Gamma, \Delta_k \cup \{A_n\} \vdash B$ by $\exists^-$ and therefore $\Gamma, \Delta_k \vdash B$.

Next, we show

$$\Gamma, \Delta_k \vdash B \to \exists_n \forall_{k' \succeq_n k}\,(B \in \Delta_{k'}), \quad \text{provided } \mathrm{FV}(B) \subseteq V_k. \tag{1.8}$$

Choose $n \geq \mathrm{lh}(k)$ such that $B = A_n$ and $\Gamma_n, \Delta_k \vdash_n A_n$. For all $k' \succeq k$, if $\mathrm{lh}(k') = n + 1$ then $A_n \in \Delta_{k'}$ (cf. the cases 2-4).

Using the sets Δ_k we can define a tree model $\mathcal{T}$ as (Ter, I_0, I_1) where Ter denotes the set of terms of the underlying language, $I_0(f)(\vec{s}) := f\vec{s}$ and

$$R^{\mathcal{T}}(\vec{s}, k) = I_1(R, k)(\vec{s}) := (R\vec{s} \in \Delta_k).$$

Obviously, $t^{\mathcal{T}}[\mathrm{id}] = t$ for all terms t.

Now write $k \Vdash B$ for $\mathcal{T}, k \Vdash B[\mathrm{id}]$. We show:

Claim. $\Gamma, \Delta_k \vdash B \leftrightarrow k \Vdash B$ *provided* $\mathrm{FV}(B) \subseteq V_k$.

The proof is by induction on B.

Case $R\vec{s}$. Assume $\mathrm{FV}(R\vec{s}) \subseteq V_k$. The following are equivalent.

$\Gamma, \Delta_k \vdash R\vec{s}$	
$\exists_n \forall_{k' \succeq_n k}\,(R\vec{s} \in \Delta_{k'})$	by (1.8) and (1.7)
$\exists_n \forall_{k' \succeq_n k}\, R^{\mathcal{T}}(\vec{s}, k')$	by definition of $\mathcal{T}$
$k \Vdash R\vec{s}$	by definition of $\Vdash$, since $t^{\mathcal{T}}[\mathrm{id}] = t$.

Case $B \vee C$. Assume $\mathrm{FV}(B \vee C) \subseteq V_k$. For the implication $\to$ let $\Gamma, \Delta_k \vdash B \vee C$. Choose an $n \geq \mathrm{lh}(k)$ such that $\Gamma_n, \Delta_k \vdash_n A_n = B \vee C$. Then, for all $k' \succeq k$ s.t. $\mathrm{lh}(k') = n$,

$$\Delta_{k'0} = \Delta_{k'} \cup \{B \vee C, B\} \quad \text{and} \quad \Delta_{k'1} = \Delta_{k'} \cup \{B \vee C, C\},$$

and therefore by induction hypothesis

$$k'0 \Vdash B \quad \text{and} \quad k'1 \Vdash C.$$

Then by definition we have $k \Vdash B \vee C$. For the reverse implication $\leftarrow$ argue as follows.

$$
\begin{array}{ll}
k \Vdash B \vee C & \\
\exists_n \forall_{k' \succeq_n k} (k' \Vdash B \vee k' \Vdash C) & \\
\exists_n \forall_{k' \succeq_n k} ((\Gamma, \Delta_{k'} \vdash B) \vee (\Gamma, \Delta_{k'} \vdash C)) & \text{by induction hypothesis} \\
\exists_n \forall_{k' \succeq_n k} (\Gamma, \Delta_{k'} \vdash B \vee C) & \\
\Gamma, \Delta_k \vdash B \vee C & \text{by (1.7).}
\end{array}
$$

Case $B \wedge C$. This is evident.

Case $B \to C$. Assume $\mathrm{FV}(B \to C) \subseteq V_k$. For $\to$ let $\Gamma, \Delta_k \vdash B \to C$. We must show $k \Vdash B \to C$, i.e.,

$$\forall_{k' \succeq k} (k' \Vdash B \to k' \Vdash C).$$

Let $k' \succeq k$ be such that $k' \Vdash B$. By induction hypothesis, it follows that $\Gamma, \Delta_{k'} \vdash B$. Hence $\Gamma, \Delta_{k'} \vdash C$ follows by assumption. Then again by induction hypothesis $k' \Vdash C$.

For $\leftarrow$ let $k \Vdash B \to C$, i.e., $\forall_{k' \succeq k} (k' \Vdash B \to k' \Vdash C)$. We show that $\Gamma, \Delta_k \vdash B \to C$, using (1.7). Choose $n \geq \mathrm{lh}(k)$ such that $B = A_n$. For all $k' \succeq_m k$ with $m := n - \mathrm{lh}(k)$ we show that $\Gamma, \Delta_{k'} \vdash B \to C$.

If $\Gamma_n, \Delta_{k'} \vdash_n A_n$, then $k' \Vdash B$ by induction hypothesis, and $k' \Vdash C$ by assumption. Hence $\Gamma_n, \Delta_{k'} \vdash C$ again by induction hypothesis and thus $\Gamma, \Delta_{k'} \vdash B \to C$.

If $\Gamma, \Delta_{k'} \nvdash_n A_n$, then by definition $\Delta_{k'1} = \Delta_{k'} \cup \{B\}$. Hence $\Gamma, \Delta_{k'1} \vdash B$, and thus $k'1 \Vdash B$ by induction hypothesis. Now $k'1 \Vdash C$ by assumption, and finally $\Gamma, \Delta_{k'1} \vdash C$ by induction hypothesis. From $\Delta_{k'1} = \Delta_{k'} \cup \{B\}$ it follows that $\Gamma, \Delta_{k'} \vdash B \to C$.

Case $\forall_x B(x)$. Assume $\mathrm{FV}(\forall_x B(x)) \subseteq V_k$. For $\to$ let $\Gamma, \Delta_k \vdash \forall_x B(x)$. Fix a term t. Then $\Gamma, \Delta_k \vdash B(t)$. Choose n such that $\mathrm{FV}(B(t)) \subseteq V_{k'}$ for all $k' \succeq_n k$. Then $\forall_{k' \succeq_n k} (\Gamma, \Delta_{k'} \vdash B(t))$, hence $\forall_{k' \succeq_n k} (k' \Vdash B(t))$ by induction hypothesis, hence $k \Vdash B(t)$ by the covering lemma. This holds for every term t, hence $k \Vdash \forall_x B(x)$.

For $\leftarrow$ assume $k \Vdash \forall_x B(x)$. Pick $k' \succeq_n k$ such that $A_m = \exists_x (\bot \to \bot)$, for $m := \mathrm{lh}(k) + n$. Then at height m we put some x_i into the variable sets: for $k' \succeq_n k$ we have $x_i \notin V_{k'}$ but $x_i \in V_{k'j}$. Clearly $k'j \Vdash B(x_i)$, hence $\Gamma, \Delta_{k'j} \vdash B(x_i)$ by induction hypothesis, hence (since at this height we consider the trivial formula $\exists_x (\bot \to \bot)$) also $\Gamma, \Delta_{k'} \vdash B(x_i)$. Since $x_i \notin V_{k'}$ we obtain $\Gamma, \Delta_{k'} \vdash \forall_x B(x)$. This holds for all $k' \succeq_n k$, hence $\Gamma, \Delta_k \vdash \forall_x B(x)$ by (1.7).

Case $\exists_x B(x)$. Assume $\mathrm{FV}(\exists_x B(x)) \subseteq V_k$. For $\to$ let $\Gamma, \Delta_k \vdash \exists_x B(x)$. Choose an $n \geq \mathrm{lh}(k)$ such that $\Gamma_n, \Delta_k \vdash_n A_n = \exists_x B(x)$. Then, for all $k' \succeq k$

with $\mathrm{lh}(k') = n$

$$\Delta_{k'0} = \Delta_{k'1} = \Delta_{k'} \cup \{\exists_x B(x), B(x_i)\}$$

where $x_i \notin V_{k'}$. Hence by induction hypothesis for $B(x_i)$ (applicable since $\mathrm{FV}(B(x_i)) \subseteq V_{k'j}$ for $j = 0, 1$)

$$k'0 \Vdash B(x_i) \quad \text{and} \quad k'1 \Vdash B(x_i).$$

It follows by definition that $k \Vdash \exists_x B(x)$.

For $\leftarrow$ assume $k \Vdash \exists_x B(x)$. Then $\forall_{k' \succeq_n k} \exists_{t \in \mathrm{Ter}} (k' \Vdash B(x)[\mathrm{id}^t_x])$ for some n, hence $\forall_{k' \succeq_n k} \exists_{t \in \mathrm{Ter}} (k' \Vdash B(t))$. For each of the finitely many $k' \succeq_n k$ pick an m such that $\forall_{k'' \succeq_m k'} (\mathrm{FV}(B(t_{k'})) \subseteq V_{k''})$. Let m_0 be the maximum of all these m. Then

$$\forall_{k'' \succeq_{m_0+n} k} \exists_{t \in \mathrm{Ter}} ((k'' \Vdash B(t)) \wedge \mathrm{FV}(B(t)) \subseteq V_{k''}).$$

The induction hypothesis for $B(t)$ yields

$$\begin{aligned}
&\forall_{k'' \succeq_{m_0+n} k} \exists_{t \in \mathrm{Ter}} (\Gamma, \Delta_{k''} \vdash B(t)) \\
&\forall_{k'' \succeq_{m_0+n} k} (\Gamma, \Delta_{k''} \vdash \exists_x B(x)) \\
&\Gamma, \Delta_k \vdash \exists_x B(x) \qquad\qquad \text{by (1.7)}
\end{aligned}$$

and this completes the proof of the claim.

Now we can finish the proof of the completeness theorem by showing that (b) implies (a). We apply (b) to the tree model $\mathcal{T}$ constructed above from Γ, the empty node $\langle\rangle$ and the assignment $\eta = \mathrm{id}$. Then $\mathcal{T}, \langle\rangle \Vdash \Gamma[\mathrm{id}]$ by the claim (since each formula in Γ is derivable from Γ). Hence $\mathcal{T}, \langle\rangle \Vdash A[\mathrm{id}]$ by (b) and therefore $\Gamma \vdash A$ by the claim again. □

Completeness of intuitionistic logic follows as a corollary.

Corollary. *Let $\Gamma \cup \{A\}$ be a set of formulas. The following propositions are equivalent.*

(a) $\Gamma \vdash_i A$.

(b) $\Gamma, \mathrm{Efq} \Vdash A$, *i.e., for all tree models $\mathcal{T}$ for intuitionistic logic, nodes k and assignments η*

$$\mathcal{T}, k \Vdash \Gamma[\eta] \to \mathcal{T}, k \Vdash A[\eta].$$ □

1.4. Soundness and Completeness of the Classical Fragment

We give a proof of completeness of classical logic relying on the completeness proof for minimal logic above.

1.4.1. Models. We define the notion of a (classical) model (or more accurately, $\mathcal{L}$-model), and what the value of a term and the meaning of a formula in a model should be. The latter definition is by induction on formulas, where in the quantifier case we need a quantifier in the definition.

For the rest of this section, fix a countable formal language $\mathcal{L}$; we do not mention the dependence on $\mathcal{L}$ in the notation. Since we deal with classical logic, we only consider formulas built without $\vee, \exists$.

DEFINITION. A *model* is a triple $\mathcal{M} = (D, I_0, I_1)$ such that

(a) D is a nonempty set;
(b) for every n-ary function symbol f, I_0 assigns to f a map $I_0(f)\colon D^n \to D$;
(c) for every n-ary relation symbol R, I_1 assigns to R an n-ary relation on D^n. In case $n = 0$, $I_1(R)$ is either true or false. We require that $I_1(\bot)$ is false.

We write $|\mathcal{M}|$ for the carrier set D of $\mathcal{M}$ and $f^{\mathcal{M}}$, $R^{\mathcal{M}}$ for the interpretations $I_0(f)$, $I_1(R)$ of the function and relation symbols. *Assignments* η and their homomorphic extensions are defined as in 1.3.1. Again we write $t^{\mathcal{M}}[\eta]$ for $\eta(t)$.

DEFINITION (Validity). For every model $\mathcal{M}$, assignment η in $|\mathcal{M}|$ and formula A such that $\mathrm{FV}(A) \subseteq \mathrm{dom}(\eta)$ we define $\mathcal{M} \models A[\eta]$ (read: A is *valid* in $\mathcal{M}$ under the assignment η) by induction on A.

$$\begin{aligned}
\mathcal{M} \models (R\vec{s})[\eta] &:= R^{\mathcal{M}}(\vec{s}^{\mathcal{M}}[\eta]),\\
\mathcal{M} \models (A \to B)[\eta] &:= ((\mathcal{M} \models A[\eta]) \to (\mathcal{M} \models B[\eta])),\\
\mathcal{M} \models (A \wedge B)[\eta] &:= ((\mathcal{M} \models A[\eta]) \wedge (\mathcal{M} \models B[\eta])),\\
\mathcal{M} \models (\forall_x A)[\eta] &:= \forall_{a \in |\mathcal{M}|}(\mathcal{M} \models A[\eta^a_x]).
\end{aligned}$$

Since $I_1(\bot)$ is false, we have $\mathcal{M} \not\models \bot[\eta]$.

1.4.2. Soundness of classical logic.

LEMMA (Coincidence). *Let $\mathcal{M}$ be a model, t a term, A a formula and η, ξ assignments in $|\mathcal{M}|$.*

(a) *If $\eta(x) = \xi(x)$ for all $x \in \mathrm{vars}(t)$, then $\eta(t) = \xi(t)$.*
(b) *If $\eta(x) = \xi(x)$ for all $x \in \mathrm{FV}(A)$, then $\mathcal{M} \models A[\eta]$ if and only if $\mathcal{M} \models A[\xi]$.*

PROOF. Induction on terms and formulas. □

LEMMA (Substitution). *Let $\mathcal{M}$ be a model, $t, r(x)$ terms, $A(x)$ a formula and η an assignment in $|\mathcal{M}|$. Then*

(a) *$\eta(r(t)) = \eta^{\eta(t)}_x(r(x))$.*
(b) *$\mathcal{M} \models A(t)$ if and only if $\mathcal{M} \models A(x)[\eta^{\eta(t)}_x]$.*

PROOF. Induction on terms and formulas. □

A model $\mathcal{M}$ is called *classical* if $\neg\neg R^{\mathcal{M}}(\vec{a}) \to R^{\mathcal{M}}(\vec{a})$ for all relation symbols R and all $\vec{a} \in |\mathcal{M}|$. We prove that every formula derivable in classical logic is valid in an arbitrary classical model.

THEOREM (Soundness of classical logic). *Let $\Gamma \cup \{A\}$ be a set of formulas such that $\Gamma \vdash_c A$. Then, if $\mathcal{M}$ is a classical model and η an assignment in $|\mathcal{M}|$, it follows that $\mathcal{M} \models \Gamma[\eta]$ implies $\mathcal{M} \models A[\eta]$.*

PROOF. Induction on derivations. We begin with the axioms in Stab and the axiom schemes $\wedge^+$, $\wedge^-$. $\mathcal{M} \models C[\eta]$ is abbreviated $\mathcal{M} \models C$ when η is known from the context.

For the stability axiom $\forall_{\vec{x}}(\neg\neg R\vec{x} \to R\vec{x})$ the claim follows from our assumption that $\mathcal{M}$ is classical, i.e., $\neg\neg R^{\mathcal{M}}(\vec{a}) \to R^{\mathcal{M}}(\vec{a})$ for all $\vec{a} \in |\mathcal{M}|$. The axioms $\wedge^+$, $\wedge^-$ are clearly valid.

This concludes the treatment of the axioms. We now consider the rules. In case of the assumption rule $u\colon A$ we have $A \in \Gamma$ and the claim is obvious.

Case $\to^+$. Assume $\mathcal{M} \models \Gamma$. We show $\mathcal{M} \models (A \to B)$. So assume in addition $\mathcal{M} \models A$. We must show $\mathcal{M} \models B$. By induction hypothesis (with $\Gamma \cup \{A\}$ instead of Γ) this clearly holds.

Case $\to^-$. Assume $\mathcal{M} \models \Gamma$. We must show $\mathcal{M} \models B$. By induction hypothesis, $\mathcal{M} \models (A \to B)$ and $\mathcal{M} \models A$. The claim follows from the definition of $\models$.

Case $\forall^+$. Assume $\mathcal{M} \models \Gamma[\eta]$ and $x \notin \mathrm{FV}(\Gamma)$. We show $\mathcal{M} \models (\forall_x A)[\eta]$, i.e., $\mathcal{M} \models A[\eta^a_x]$ for an arbitrary $a \in |\mathcal{M}|$. We have

$$\begin{array}{ll} \mathcal{M} \models \Gamma[\eta^a_x] & \text{by the coincidence lemma, since } x \notin \mathrm{FV}(\Gamma) \\ \mathcal{M} \models A[\eta^a_x] & \text{by induction hypothesis.} \end{array}$$

Case $\forall^-$. Let $\mathcal{M} \models \Gamma[\eta]$. We show that $\mathcal{M} \models A(t)[\eta]$. This follows from

$$\begin{array}{ll} \mathcal{M} \models (\forall_x A(x))[\eta] & \text{by induction hypothesis} \\ \mathcal{M} \models A(x)[\eta^{\eta(t)}_x] & \text{by definition} \\ \mathcal{M} \models A(t)[\eta] & \text{by the substitution lemma.} \end{array}$$

This concludes the proof. □

1.4.3. Completeness of classical logic. We give a constructive analysis of the completeness of classical logic by using, in the metatheory below, constructively valid arguments only, mentioning explicitly any assumptions which go beyond. When dealing with the classical fragment we of course need to restrict to classical models. The only non-constructive principle

will be the use of the *axiom of dependent choice* for the weak existential quantifier

$$\tilde{\exists}_x A(0,x) \to \forall_{n,x}(A(n,x) \to \tilde{\exists}_y A(n+1,y)) \to \tilde{\exists}_f \forall_n A(n,fn).$$

Recall that we only consider formulas without $\vee, \exists$.

THEOREM (Completeness of classical logic). *Let $\Gamma \cup \{A\}$ be a set of formulas. Assume that for all classical models $\mathcal{M}$ and assignments η,*

$$\mathcal{M} \models \Gamma[\eta] \to \mathcal{M} \models A[\eta].$$

Then there must exist a derivation of A from $\Gamma \cup \mathrm{Stab}$.

PROOF. Since "there must exist a derivation" expresses the weak existential quantifier in the metalanguage, we need to prove a contradiction from the assumption $\Gamma, \mathrm{Stab} \nvdash A$.

By the completeness theorem for minimal logic, there must be a tree model $\mathcal{T} = (\mathrm{Ter}, I_0, I_1)$ on the complete binary tree T_{01} and a node l_0 such that $l_0 \Vdash \Gamma, \mathrm{Stab}$ and $l_0 \nVdash A$.

Call a node k *consistent* if $k \nVdash \bot$, and *stable* if $k \Vdash \mathrm{Stab}$. We prove

$$k \nVdash B \to \tilde{\exists}_{k' \succeq k}(k' \Vdash \neg B \wedge k' \nVdash \bot) \qquad (k \text{ stable}). \tag{1.9}$$

Let k be a stable node, and B a formula (without $\vee$, $\exists$). Then $\mathrm{Stab} \vdash \neg\neg B \to B$ by the stability theorem, and therefore $k \Vdash \neg\neg B \to B$. Hence from $k \nVdash B$ we obtain $k \nVdash \neg\neg B$. By a remark in 1.3.4 this implies that $\neg\forall_{k' \succeq k}(k' \Vdash \neg B \to k' \Vdash \bot)$, which proves (1.9).

Let α be a branch in the underlying tree T_{01}. We define

$$\begin{aligned} \alpha \Vdash A \qquad &:= \tilde{\exists}_{k \in \alpha}(k \Vdash A), \\ \alpha \text{ is consistent} &:= \alpha \nVdash \bot, \\ \alpha \text{ is stable} \qquad &:= \tilde{\exists}_{k \in \alpha}(k \Vdash \mathrm{Stab}). \end{aligned}$$

Note that from $\alpha \Vdash \vec{A}$ and $\vdash \vec{A} \to B$ it follows that $\alpha \Vdash B$. To see this, consider $\alpha \Vdash \vec{A}$. Then $k \Vdash \vec{A}$ for a $k \in \alpha$, since α is linearly ordered. From $\vdash \vec{A} \to B$ it follows that $k \Vdash B$, i.e., $\alpha \Vdash B$.

A branch α is *generic* (in the sense that it generates a classical model) if it is consistent and stable, if in addition for all formulas B

$$(\alpha \Vdash B) \,\tilde{\vee}\, (\alpha \Vdash \neg B), \tag{1.10}$$

and if for all formulas $\forall_{\vec{y}} B(\vec{y})$ with $B(\vec{y})$ not a universal formula,

$$\forall_{\vec{s} \in \mathrm{Ter}}(\alpha \Vdash B(\vec{s})) \to \alpha \Vdash \forall_{\vec{y}} B(\vec{y}). \tag{1.11}$$

For a branch α, we define a classical model $\mathcal{M}^\alpha = (\mathrm{Ter}, I_0, I_1^\alpha)$ as

$$I_1^\alpha(R)(\vec{s}) := \tilde{\exists}_{k \in \alpha} I_1(R,k)(\vec{s}) \qquad (R \neq \bot).$$

Since $\tilde{\exists}$ is used in this definition, $\mathcal{M}^\alpha$ is stable.

We show that for every generic branch α and formula B (without $\vee$, $\exists$)

$$\alpha \Vdash B \leftrightarrow \mathcal{M}^\alpha \models B. \tag{1.12}$$

The proof is by induction on the logical complexity of B.

Case $R\vec{s}$ with $R \neq \bot$. Then (1.12) holds for all α.

Case $\bot$. We have $\alpha \nVdash \bot$ since α is consistent.

Case $B \to C$. Let $\alpha \Vdash B \to C$ and $\mathcal{M}^\alpha \models B$. We must show that $\mathcal{M}^\alpha \models C$. Note that $\alpha \Vdash B$ by induction hypothesis, hence $\alpha \Vdash C$, hence $\mathcal{M}^\alpha \models C$ again by induction hypothesis. Conversely let $\mathcal{M}^\alpha \models B \to C$. Clearly $(\mathcal{M}^\alpha \models B) \mathbin{\tilde{\vee}} (\mathcal{M}^\alpha \not\models B)$. If $\mathcal{M}^\alpha \models B$, then $\mathcal{M}^\alpha \models C$. Hence $\alpha \Vdash C$ by induction hypothesis and therefore $\alpha \Vdash B \to C$. If $\mathcal{M}^\alpha \not\models B$ then $\alpha \nVdash B$ by induction hypothesis. Hence $\alpha \Vdash \neg B$ by (1.10) and therefore $\alpha \Vdash B \to C$, since α is stable (and $\vdash (\neg\neg C \to C) \to \bot \to C$). [Note that for this argument to be contructively valid one needs to observe that the formula $\alpha \Vdash B \to C$ is a negation, and therefore we can argue by the case distinction based on $\tilde{\vee}$. This is because, with $P_1 := \mathcal{M}^\alpha \models B$, $P_2 := \mathcal{M}^\alpha \not\models B$ and $Q := \alpha \Vdash B \to C$, the formula $(P_1 \mathbin{\tilde{\vee}} P_2) \to (P_1 \to Q) \to (P_2 \to Q) \to Q$ is derivable in minimal logic.]

Case $B \wedge C$. Easy.

Case $\forall_{\vec{y}} B(\vec{y})$ ($\vec{y}$ not empty) where $B(\vec{y})$ is not a universal formula. The following are equivalent.

$$\begin{aligned}
&\alpha \Vdash \forall_{\vec{y}} B(\vec{y}) \\
&\forall_{\vec{s} \in \mathrm{Ter}}(\alpha \Vdash B(\vec{s})) && \text{by (1.11)} \\
&\forall_{\vec{s} \in \mathrm{Ter}}(\mathcal{M}^\alpha \models B(\vec{s})) && \text{by induction hypothesis} \\
&\mathcal{M}^\alpha \models \forall_{\vec{y}} B(\vec{y}).
\end{aligned}$$

This concludes the proof of (1.12).

Next we show that for every consistent and stable node k there must be a generic branch containing k:

$$k \nVdash \bot \to k \Vdash \mathrm{Stab} \to \tilde{\exists}_\alpha(\alpha \text{ generic} \wedge k \in \alpha). \tag{1.13}$$

For the proof, let $A_0, A_1, \ldots$ enumerate all formulas. We define a sequence $k = k_0 \preceq k_1 \preceq k_2 \ldots$ of consistent stable nodes by dependent choice. Let $k_0 := k$. Assume that k_n is defined. We write A_n in the form $\forall_{\vec{y}} B(\vec{y})$ (with $\vec{y}$ possibly empty) where B is not a universal formula. In case $k_n \Vdash \forall_{\vec{y}} B(\vec{y})$ let $k_{n+1} := k_n$. Otherwise we have $k_n \nVdash B(\vec{s})$ for some $\vec{s}$, and by (1.9) there must be a consistent node $k' \succeq k_n$ such that $k' \Vdash \neg B(\vec{s})$. Let $k_{n+1} := k'$. Since $k_n \preceq k_{n+1}$, the node k_{n+1} is stable.

Let $\alpha := \{\, l \mid \exists_n(l \preceq k_n) \,\}$, hence $k \in \alpha$. We show that α is generic. Clearly α is consistent and stable. We now prove both (1.10) and (1.11).

Let $C = \forall_{\vec{y}} B(\vec{y})$ (with $\vec{y}$ possibly empty) where $B(\vec{y})$ is not a universal formula, and choose n such that $C = A_n$. In case $k_n \Vdash \forall_{\vec{y}} B(\vec{y})$ we are done. Otherwise by construction $k_{n+1} \Vdash \neg B(\vec{s})$ for some $\vec{s}$. For (1.10) we get $k_{n+1} \Vdash \neg\forall_{\vec{y}} B(\vec{y})$ since $\vdash \forall_{\vec{y}} B(\vec{y}) \to B(\vec{s})$, and (1.11) follows from the consistency of α. This concludes the proof of (1.13).

Now we can finalize the completeness proof. Recall that $l_0 \Vdash \Gamma, \mathrm{Stab}$ and $l_0 \not\Vdash A$. Since $l_0 \not\Vdash A$ and l_0 is stable, (1.9) yields a consistent node $k \succeq l_0$ such that $k \Vdash \neg A$. Evidently, k is stable as well. By (1.13) there must be a generic branch α such that $k \in \alpha$. Since $k \Vdash \neg A$ it follows that $\alpha \Vdash \neg A$, hence $\mathcal{M}^\alpha \models \neg A$ by (1.12). Moreover, $\alpha \Vdash \Gamma$, thus $\mathcal{M}^\alpha \models \Gamma$ by (1.12). This contradicts our assumption. □

1.4.4. Compactness and Löwenheim-Skolem theorems. Among the many important corollaries of the completeness theorem the compactness and Löwenheim-Skolem theorems stand out as particularly important. A set Γ of formulas is *consistent* if $\Gamma \nvdash_c \bot$, and *satisfiable* if there is (in the weak sense) a classical model $\mathcal{M}$ and an assignment η in $|\mathcal{M}|$ such that $\mathcal{M} \models \Gamma[\eta]$.

Corollary. *Let Γ be a set of formulas.*

(a) *If Γ is consistent, then Γ is satisfiable.*
(b) *(Compactness). If each finite subset of Γ is satisfiable, Γ is satisfiable.*

Proof. (a). Assume $\Gamma \nvdash_c \bot$ and that for all classical models $\mathcal{M}$ we have $\mathcal{M} \not\models \Gamma$, i.e., $\mathcal{M} \models \Gamma$ implies $\mathcal{M} \models \bot$. Then the completeness theorem yields a contradiction.

(b). Otherwise by the completeness theorem there must be a derivation of $\bot$ from $\Gamma \cup \mathrm{Stab}$, hence also from $\Gamma_0 \cup \mathrm{Stab}$ for some finite subset $\Gamma_0 \subseteq \Gamma$. This contradicts the assumption that Γ_0 is satisfiable. □

Corollary (Löwenheim and Skolem). *Let Γ be a set of formulas (we assume that $\mathcal{L}$ is countable). If Γ is satisfiable, then Γ is satisfiable in a model with a countably infinite carrier set.*

Proof. Assume that Γ is not satisfiable in a countable model. Then by the completeness theorem $\Gamma \cup \mathrm{Stab} \vdash \bot$. Therefore by the soundness theorem Γ cannot be satisfiable. □

CHAPTER 2

Model Theory

Model theory is an established branch of mathematical logic. It uses tools from logic to study questions in algebra. In model theory it is common to disregard the distinction between strong and weak existential quantifiers; we shall do the same in the present chapter. Also, the restriction to countable languages that we have maintained until now is given up. Moreover one makes free use of other concepts and axioms from set theory like the *axiom of choice* (for the weak existential quantifier), most often in the form of *Zorn's lemma*.

2.1. Ultraproducts

2.1.1. Filters and ultrafilters. Let $M \neq \emptyset$ be a set. $F \subseteq \mathcal{P}(M)$ is called *filter* on M if

(a) $M \in F$ and $\emptyset \notin F$;
(b) if $X \in F$ and $X \subseteq Y \subseteq M$, then $Y \in F$;
(c) $X, Y \in F$ entails $X \cap Y \in F$.

F is called *ultrafilter* if for all $X \in \mathcal{P}(M)$

$$X \in F \text{ or } M \setminus X \in F.$$

The intuition here is that the elements X of a filter F are considered to be "big". For instance, for M infinite the set $F = \{ X \subseteq M \mid M \setminus X \text{ finite} \}$ is a filter (called *Fréchet*-filter).

LEMMA. *Suppose F is an ultrafilter and $X \cup Y \in F$. Then $X \in F$ or $Y \in F$.*

PROOF. If both X and Y are not in F, then $M \setminus X$ and $M \setminus Y$ are in F, hence also $(M \setminus X) \cap (M \setminus Y)$, which is $M \setminus (X \cup Y)$. This contradicts the assumption $X \cup Y \in F$. □

Let $M \neq \emptyset$ be a set and $S \subseteq \mathcal{P}(M)$. S has the *finite intersection property* if $X_1 \cap \cdots \cap X_n \neq \emptyset$ for all $X_1, \ldots, X_n \in S$ and all $n \in \mathbb{N}$.

LEMMA. *If S has the finite intersection property, then there exists a filter F on M such that $F \supseteq S$.*

PROOF. $F := \{\, X \mid X \supseteq X_1 \cap \cdots \cap X_n \text{ for some } X_1, \ldots, X_n \in S \,\}$. □

THEOREM (Ultrafilter). *Let $M \neq \emptyset$ be a set and F a filter on M. Then there is an ultrafilter U on M such that $U \supseteq F$.*

PROOF. By Zorn's lemma (which will be proved from the axiom of choice later, in the chapter on set theory), there is a maximal filter U with $F \subseteq U$. We claim that U is an ultrafilter. So let $X \subseteq M$ and assume $X \notin U$ and $M \setminus X \notin U$. Since U is maximal, $U \cup \{X\}$ cannot have the finite intersection property; hence there is a $Y \in U$ such that $Y \cap X = \emptyset$. Similary we obtain $Z \in U$ such that $Z \cap (M \setminus X) = \emptyset$. But then $Y \cap Z = \emptyset$, a contradiction. □

2.1.2. Products and ultraproducts. Let $I \neq \emptyset$ be a set and $D_i \neq \emptyset$ sets for $i \in I$. Let

$$\prod_{i \in I} D_i := \{\, \alpha \mid \alpha \text{ is a function, } \mathrm{dom}(\alpha) = I \text{ and } \alpha(i) \in D_i \text{ for all } i \in I \,\}.$$

Observe that, by the *axiom of choice*, $\prod_{i\in I} D_i \neq \emptyset$. We write $\alpha \in \prod_{i\in I} D_i$ as $\langle\, \alpha(i) \mid i \in I \,\rangle$.

Now let $I \neq \emptyset$ be a set, F a filter on I and $\mathcal{M}_i$ models for $i \in I$. Then the *F-product* $\mathcal{M} = \prod_{i\in I}^F \mathcal{M}_i$ is defined by

(a) $|\mathcal{M}| := \prod_{i\in I} |\mathcal{M}_i|$ (notice that $|\mathcal{M}| \neq \emptyset$).

(b) for an n-ary relation symbol R and $\alpha_1, \ldots, \alpha_n \in |\mathcal{M}|$ let

$$R^{\mathcal{M}}(\alpha_1, \ldots, \alpha_n) := (\{\, i \in I \mid R^{\mathcal{M}_i}(\alpha_1(i), \ldots, \alpha_n(i)) \,\} \in F).$$

(c) for an n-ary function symbol f and $\alpha_1, \ldots, \alpha_n \in |\mathcal{M}|$ let

$$f^{\mathcal{M}}(\alpha_1, \ldots, \alpha_n) := \langle\, f^{\mathcal{M}_i}(\alpha_1(i), \ldots, \alpha_n(i)) \mid i \in I \,\rangle.$$

For an ultrafilter U we call $\mathcal{M} = \prod_{i\in I}^U \mathcal{M}_i$ the *U-ultraproduct* of the $\mathcal{M}_i$.

THEOREM (Fundamental theorem on ultraproducts, Łoś (1955)). *Let $\mathcal{M} = \prod_{i\in I}^U \mathcal{M}_i$ be a U-ultraproduct, A a formula and η an assignment in $|\mathcal{M}|$. Then*

$$\mathcal{M} \models A[\eta] \leftrightarrow \{\, i \in I \mid \mathcal{M}_i \models A[\eta_i] \,\} \in U,$$

where η_i is the assignment induced by $\eta_i(x) = \eta(x)(i)$ for $i \in I$.

PROOF. We first prove a similar property for terms.

$$t^{\mathcal{M}}[\eta] = \langle\, t^{\mathcal{M}_i}[\eta_i] \mid i \in I \,\rangle. \tag{2.1}$$

The proof is by induction on t. For a variable the claim follows from the definition. *Case* $f(t_1, \ldots, t_n)$. For simplicity assume $n = 1$; so we consider ft. We obtain

$$\begin{aligned}(ft)^{\mathcal{M}}[\eta] &= f^{\mathcal{M}}(t^{\mathcal{M}}[\eta]) \\ &= f^{\mathcal{M}}\langle\, t^{\mathcal{M}_i}[\eta_i] \mid i \in I \,\rangle \quad \text{by induction hypothesis}\end{aligned}$$

$$= \langle\, (ft)^{\mathcal{M}_i}[\eta_i] \mid i \in I \,\rangle.$$

Case $R(t_1, \ldots, t_n)$. For simplicity assume $n = 1$; so consider Rt. We obtain

$$\begin{aligned}
\mathcal{M} \models Rt[\eta] &\leftrightarrow R^{\mathcal{M}}(t^{\mathcal{M}}[\eta]) \\
&\leftrightarrow \{\, i \in I \mid R^{\mathcal{M}_i}(t^{\mathcal{M}}[\eta](i)) \,\} \in U \\
&\leftrightarrow \{\, i \in I \mid R^{\mathcal{M}_i}(t^{\mathcal{M}_i}[\eta_i]) \,\} \in U \qquad \text{by (2.1)} \\
&\leftrightarrow \{\, i \in I \mid \mathcal{M}_i \models Rt[\eta_i] \,\} \in U.
\end{aligned}$$

Case $A \to B$.

$\mathcal{M} \models (A \to B)[\eta]$
$\leftrightarrow$ if $\mathcal{M} \models A[\eta]$, then $\mathcal{M} \models B[\eta]$
$\leftrightarrow$ if $\{\, i \in I \mid \mathcal{M}_i \models A[\eta_i] \,\} \in U$, then $\{\, i \in I \mid \mathcal{M}_i \models B[\eta_i] \,\} \in U$
by induction hypothesis
$\leftrightarrow \{\, i \in I \mid \mathcal{M}_i \models A[\eta_i] \,\} \notin U$ or $\{\, i \in I \mid \mathcal{M}_i \models B[\eta_i] \,\} \in U$
$\leftrightarrow \{\, i \in I \mid \mathcal{M}_i \models \neg A[\eta_i] \,\} \in U$ or $\{\, i \in I \mid \mathcal{M}_i \models B[\eta_i] \,\} \in U$
for U is an ultrafilter
$\leftrightarrow \{\, i \in I \mid \mathcal{M}_i \models (A \to B)[\eta_i] \,\} \in U.$

The case $A \wedge B$ is easy.
Case $\forall_x A$.

$\mathcal{M} \models (\forall_x A)[\eta]$
$\leftrightarrow \forall_{\alpha \in |\mathcal{M}|}(\mathcal{M} \models A[\eta_x^\alpha])$
$\leftrightarrow \forall_{\alpha \in |\mathcal{M}|}(\{\, i \in I \mid \mathcal{M}_i \models A[(\eta_i)_x^{\alpha(i)}] \,\} \in U)$ by induction hypothesis
$\leftrightarrow^{(*)} \{\, i \in I \mid \forall_{a \in |\mathcal{M}_i|}(\mathcal{M}_i \models A[(\eta_i)_x^a]) \,\} \in U$ see below
$\leftrightarrow \{\, i \in I \mid \mathcal{M}_i \models (\forall_x A)[\eta_i] \,\} \in U.$

It remains to the equivalence marked $(*)$. Let

$$X := \{\, i \in I \mid \forall_{a \in |\mathcal{M}_i|}(\mathcal{M}_i \models A[(\eta_i)_x^a]) \,\}$$

and $Y_\alpha := \{\, i \in I \mid \mathcal{M}_i \models A[(\eta_i)_x^{\alpha(i)}] \,\}$ for $\alpha \in |\mathcal{M}|$.

$\leftarrow$. Let $\alpha \in |\mathcal{M}|$ and $X \in U$. Clearly $X \subseteq Y_\alpha$, hence also $Y_\alpha \in U$.

$\rightarrow$. Let $Y_\alpha \in U$ for all α. Assume $X \notin U$. Since U is an ultrafilter,

$$I \setminus X = \{\, i \in I \mid \exists_{a \in |\mathcal{M}_i|}(\mathcal{M}_i \not\models A[(\eta_i)_x^a]) \,\} \in U.$$

We choose by the axiom of choice an $\alpha_0 \in |\mathcal{M}|$ such that

$$\alpha_0(i) = \begin{cases} \text{some } a \in |\mathcal{M}_i| \text{ such that } \mathcal{M}_i \not\models A[(\eta_i)_x^a] & \text{if } i \in I \setminus X, \\ \text{an arbitrary} \in |\mathcal{M}_i| & \text{otherwise.} \end{cases}$$

Then $Y_{\alpha_0} \cap (I \setminus X) = \emptyset$, contradicting $Y_{\alpha_0}, I \setminus X \in U$. □

If we choose $\mathcal{M}_i = \mathcal{N}$ constant, then $\mathcal{M} = \prod_{i\in I}^U \mathcal{N}$ satisfies the same closed formulas as $\mathcal{N}$ (such models will be called *elementary equivalent*; the notation is $\mathcal{M} \equiv \mathcal{N}$). $\prod_{i\in I}^U \mathcal{N}$ is called an *ultrapower* of $\mathcal{N}$.

2.1.3. General compactness and completeness. Recall that the underlying language may be uncountable.

Corollary (General compactness theorem). *Let Γ be a set of formulas. If every finite subset of Γ is satisfiable, then so is Γ.*

Proof. Let $I := \{\, i \subseteq \Gamma \mid i \text{ finite} \,\}$. For $i \in I$ let $\mathcal{M}_i$ be a model of i under the assignment η_i. For $A \in \Gamma$ let $Z_A := \{\, i \in I \mid A \in i \,\} = \{\, i \subseteq \Gamma \mid i \text{ finite and } A \in i \,\}$. Then $F := \{\, Z_A \mid A \in \Gamma \,\}$ has the finite intersection property (for $\{A_1, \dots, A_n\} \in Z_{A_1} \cap \dots \cap Z_{A_n}$). By the Ultrafilter Theorem in 2.1.1 there is an ultrafilter U on I such that $F \subseteq U$. We consider the ultraproduct $\mathcal{M} := \prod_{i\in I}^U \mathcal{M}_i$ and the product assigment η defined by $\eta(x)(i) := \eta_i(x)$, and show $\mathcal{M} \models \Gamma[\eta]$. So let $A \in \Gamma$. By Łoś's theorem it suffices to show

$$X_A := \{\, i \in I \mid \mathcal{M}_i \models A[\eta_i] \,\} \in U.$$

But this follows from $Z_A \subseteq X_A$ and $Z_A \in F \subseteq U$. □

For every set Γ of formulas let $L(\Gamma)$ be the set of all function and relation symbols occurring in Γ. If $\mathcal{L}'$ is a sublanguage of $\mathcal{L}$, $\mathcal{M}'$ an $\mathcal{L}'$-model and $\mathcal{M}$ an $\mathcal{L}$-model, then $\mathcal{M}$ is called an *expansion* of $\mathcal{M}'$ (and $\mathcal{M}'$ a *reduct* of $\mathcal{M}$) if $|\mathcal{M}'| = |\mathcal{M}|$, $f^{\mathcal{M}'} = f^{\mathcal{M}}$ for all function symbols and $R^{\mathcal{M}'} = R^{\mathcal{M}}$ for all relation symbols in the language $\mathcal{L}'$. The (uniquely determined) $\mathcal{L}'$-reduct of $\mathcal{M}$ is denoted by $\mathcal{M}{\restriction}\mathcal{L}'$. If $\mathcal{M}$ is an expansion of $\mathcal{M}'$ and η an assignment in $|\mathcal{M}'|$, then clearly $t^{\mathcal{M}'}[\eta] = t^{\mathcal{M}}[\eta]$ for every $\mathcal{L}'$-term t and $\mathcal{M}' \models A[\eta]$ if and only if $\mathcal{M} \models A[\eta]$, for every $\mathcal{L}'$-formula A.

Corollary (General completeness theorem). *Let $\Gamma \cup \{A\}$ be a set of formulas. Assume that for all models $\mathcal{M}$ and assignments η,*

$$\mathcal{M} \models \Gamma[\eta] \to \mathcal{M} \models A[\eta].$$

Then $\Gamma \vdash_c A$.

Proof. By assumption $\Gamma \cup \{\neg A\}$ is not satisfiable. Hence by the general compactness theorem there is a finite subset $\Gamma' \subseteq \Gamma$ such that already $\Gamma' \cup \{\neg A\}$ is not satisfiable. Let $\mathcal{L}$ be the underlying (possibly uncountable) language, and $\mathcal{L}'$ the countable sublanguage containing only function and relation symbols from Γ'. By the remark above $\Gamma' \cup \{\neg A\}$ is not satisfiable w.r.t. $\mathcal{L}'$ as well. By the completeness theorem for countable languages we obtain $\Gamma' \vdash_c A$, hence $\Gamma \vdash_c A$. □

2.2. Complete Theories and Elementary Equivalence

We assume in this section that our underlying language $\mathcal{L}$ contains a binary relation symbol $=$.

2.2.1. Equality axioms. The set $\mathrm{Eq}_\mathcal{L}$ of $\mathcal{L}$-equality axioms consists of (the universal closures of)

$$x = x \quad \text{(reflexivity)},$$
$$x = y \to y = x \quad \text{(symmetry)},$$
$$x = y \to y = z \to x = z \quad \text{(transitivity)},$$
$$x_1 = y_1 \to \cdots \to x_n = y_n \to f(x_1, \ldots, x_n) = f(y_1, \ldots, y_n),$$
$$x_1 = y_1 \to \cdots \to x_n = y_n \to R(x_1, \ldots, x_n) \to R(y_1, \ldots, y_n),$$

for all n-ary function symbols f and relation symbols R of the language $\mathcal{L}$.

Lemma (Equality). (a) $\mathrm{Eq}_\mathcal{L} \vdash t = s \to r(t) = r(s)$.
(b) $\mathrm{Eq}_\mathcal{L} \vdash t = s \to (A(t) \leftrightarrow A(s))$.

Proof. (a). Induction on r. (b). Induction on A. □

An $\mathcal{L}$-model $\mathcal{M}$ satisfies the equality axioms if and only if $=^\mathcal{M}$ is a *congruence relation* (i.e., an equivalence relation compatible with the functions and relations of $\mathcal{M}$). In this section we assume that all $\mathcal{L}$-models $\mathcal{M}$ considered satisfy the equality axioms. The coincidence lemma then also holds with $=^\mathcal{M}$ instead of $=$:

Lemma (Coincidence). *Let η and ξ be assignments in $|\mathcal{M}|$ such that $\mathrm{dom}(\eta) = \mathrm{dom}(\xi)$ and $\eta(x) =^\mathcal{M} \xi(x)$ for all $x \in \mathrm{dom}(\eta)$. Then*
(a) *$t^\mathcal{M}[\eta] =^\mathcal{M} t^\mathcal{M}[\xi]$ if $\mathrm{vars}(t) \subseteq \mathrm{dom}(\eta)$ and*
(b) *$\mathcal{M} \models A[\eta] \leftrightarrow \mathcal{M} \models A[\xi]$ if $\mathrm{FV}(A) \subseteq \mathrm{dom}(\eta)$.*

Proof. Induction on t and A, respectively. □

2.2.2. Cardinality of models. Let $\mathcal{M}/{=^\mathcal{M}}$ be the *quotient model*, whose carrier set consists of congruence classes. We call a model $\mathcal{M}$ *infinite* (*countable*, *of cardinality n*) if $|\mathcal{M}/{=^\mathcal{M}}|$ is infinite (countable, of cardinality n). By an *axiom system* Γ we mean a set of closed formulas such that $\mathrm{Eq}_{L(\Gamma)} \subseteq \Gamma$. A *model* of an axiom system Γ is an $\mathcal{L}$-model $\mathcal{M}$ such that $L(\Gamma) \subseteq \mathcal{L}$ and $\mathcal{M} \models \Gamma$. For sets Γ of closed formulas we write

$$\mathrm{Mod}_\mathcal{L}(\Gamma) := \{\, \mathcal{M} \mid \mathcal{M} \text{ is an } \mathcal{L}\text{-model and } \mathcal{M} \models \Gamma \cup \mathrm{Eq}_\mathcal{L} \,\}.$$

Clearly Γ is satisfiable if and only if Γ has an $\mathcal{L}$-model.

Theorem. *If an axiom system has arbitrarily large finite models, then it has an infinite model.*

PROOF. Let Γ be such an axiom system. Suppose $x_0, x_1, x_2, \dots$ are distinct variables and

$$\Gamma' := \Gamma \cup \{\, x_i \neq x_j \mid i, j \in \mathbb{N} \text{ such that } i < j \,\}.$$

By assumption every finite subset of Γ' is satisfiable, hence by the general compactness theorem so is Γ'. Then we have $\mathcal{M}$ and η such that $\mathcal{M} \models \Gamma'[\eta]$ and therefore $\eta(x_i) \neq^{\mathcal{M}} \eta(x_j)$ for $i < j$. Hence $\mathcal{M}$ is infinite. □

2.2.3. Complete theories, elementary equivalence. Let $\overline{\mathcal{L}}$ be the set of all closed $\mathcal{L}$-formulas. By a *theory* T we mean an axiom system closed under $\vdash_c$, that is, $\mathrm{Eq}_{L(T)} \subseteq T$ and

$$T = \{\, A \in \overline{L(T)} \mid T \vdash_c A \,\}.$$

A theory T is called *complete* if for every formula $A \in \overline{L(T)}$, $T \vdash_c A$ or $T \vdash_c \neg A$.

For every $\mathcal{L}$-model $\mathcal{M}$ (satisfying the equality axioms) the set of all closed $\mathcal{L}$-formulas A such that $\mathcal{M} \models A$ clearly is a theory; it is called the *theory of* $\mathcal{M}$ and denoted by $\mathrm{Th}(\mathcal{M})$.

Two $\mathcal{L}$-models $\mathcal{M}$ and $\mathcal{M}'$ are called *elementarily equivalent* (written $\mathcal{M} \equiv \mathcal{M}'$) if $\mathrm{Th}(\mathcal{M}) = \mathrm{Th}(\mathcal{M}')$. Two $\mathcal{L}$-models $\mathcal{M}$ and $\mathcal{M}'$ are called *isomorphic* (written $\mathcal{M} \cong \mathcal{M}'$) if there is a map $\pi\colon |\mathcal{M}| \to |\mathcal{M}'|$ inducing a bijection between $|\mathcal{M}/{=^{\mathcal{M}}}|$ and $|\mathcal{M}'/{=^{\mathcal{M}'}}|$, that is,

$$\forall_{a,b \in |\mathcal{M}|}(a =^{\mathcal{M}} b \leftrightarrow \pi(a) =^{\mathcal{M}'} \pi(b)),$$
$$\forall_{a' \in |\mathcal{M}'|} \exists_{a \in |\mathcal{M}|}(\pi(a) =^{\mathcal{M}'} a'),$$

such that for all $a_1, \dots, a_n \in |\mathcal{M}|$

$$\pi(f^{\mathcal{M}}(a_1, \dots, a_n)) =^{\mathcal{M}'} f^{\mathcal{M}'}(\pi(a_1), \dots, \pi(a_n)),$$
$$R^{\mathcal{M}}(a_1, \dots, a_n) \leftrightarrow R^{\mathcal{M}'}(\pi(a_1), \dots, \pi(a_n))$$

for all n-ary function symbols f and relation symbols R of the language $\mathcal{L}$.

We collect some simple properties of the notions of the theory of a model $\mathcal{M}$ and of elementary equivalence.

LEMMA. (a) $\mathrm{Th}(\mathcal{M})$ *is complete.*

(b) *If* Γ *is an axiom system such that* $L(\Gamma) \subseteq \mathcal{L}$*, then*

$$\{\, A \in \overline{\mathcal{L}} \mid \Gamma \cup \mathrm{Eq}_{\mathcal{L}} \vdash_c A \,\} = \bigcap \{\, \mathrm{Th}(\mathcal{M}) \mid \mathcal{M} \in \mathrm{Mod}_{\mathcal{L}}(\Gamma) \,\}.$$

(c) $\mathcal{M} \equiv \mathcal{M}' \leftrightarrow \mathcal{M} \models \mathrm{Th}(\mathcal{M}')$.

(d) *If* $\mathcal{L}$ *is countable, then for every* $\mathcal{L}$*-model* $\mathcal{M}$ *there is a countable* $\mathcal{L}$*-model* $\mathcal{M}'$ *such that* $\mathcal{M} \equiv \mathcal{M}'$.

PROOF. (a). Let $\mathcal{M}$ be an $\mathcal{L}$-model and $A \in \overline{\mathcal{L}}$. Then $\mathcal{M} \models A$ or $\mathcal{M} \models \neg A$, hence $\mathrm{Th}(\mathcal{M}) \vdash_c A$ or $\mathrm{Th}(\mathcal{M}) \vdash_c \neg A$.

(b). For all $A \in \overline{\mathcal{L}}$ we have

$$\begin{aligned}\Gamma \cup \mathrm{Eq}_{\mathcal{L}} \vdash_c A &\leftrightarrow \text{for all } \mathcal{L}\text{-models } \mathcal{M},\ (\mathcal{M} \models \Gamma \to \mathcal{M} \models A)\\ &\leftrightarrow \text{for all } \mathcal{L}\text{-models } \mathcal{M},\ (\mathcal{M} \in \mathrm{Mod}_{\mathcal{L}}(\Gamma) \to A \in \mathrm{Th}(\mathcal{M}))\\ &\leftrightarrow A \in \bigcap\{\, \mathrm{Th}(\mathcal{M}) \mid \mathcal{M} \in \mathrm{Mod}_{\mathcal{L}}(\Gamma)\,\}.\end{aligned}$$

(c). For $\to$ assume $\mathcal{M} \equiv \mathcal{M}'$ and $A \in \mathrm{Th}(\mathcal{M}')$. Then $\mathcal{M}' \models A$, hence $\mathcal{M} \models A$. For $\leftarrow$ assume $\mathcal{M} \models \mathrm{Th}(\mathcal{M}')$. Then clearly $\mathrm{Th}(\mathcal{M}') \subseteq \mathrm{Th}(\mathcal{M})$. For the converse inclusion let $A \in \mathrm{Th}(\mathcal{M})$. If $A \notin \mathrm{Th}(\mathcal{M}')$, then $\neg A \in \mathrm{Th}(\mathcal{M}')$ by (a) and hence $\mathcal{M} \models \neg A$, contradicting $A \in \mathrm{Th}(\mathcal{M})$.

(d). Let $\mathcal{L}$ be countable and $\mathcal{M}$ an $\mathcal{L}$-model. Then $\mathrm{Th}(\mathcal{M})$ is satisfiable and therefore by the theorem of Löwenheim and Skolem possesses a satisfying $\mathcal{L}$-model $\mathcal{M}'$ with the countable carrier set $\mathrm{Ter}_{\mathcal{L}}$. By (c), $\mathcal{M} \equiv \mathcal{M}'$. □

Moreover, we can characterize complete theories as follows:

THEOREM. *Let T be a theory and $\mathcal{L} = L(T)$. Then the following are equivalent.*

(a) *T is complete.*
(b) *For every model $\mathcal{M} \in \mathrm{Mod}_{\mathcal{L}}(T)$, $\mathrm{Th}(\mathcal{M}) = T$.*
(c) *Any two models $\mathcal{M}, \mathcal{M}' \in \mathrm{Mod}_{\mathcal{L}}(T)$ are elementarily equivalent.*

PROOF. (a) $\to$ (b). Let T be complete and $\mathcal{M} \in \mathrm{Mod}_{\mathcal{L}}(T)$. Then $\mathcal{M} \models T$, hence $T \subseteq \mathrm{Th}(\mathcal{M})$. For the converse assume $A \in \mathrm{Th}(\mathcal{M})$. Then $\neg A \notin \mathrm{Th}(\mathcal{M})$, hence $\neg A \notin T$ and therefore $A \in T$.

(b) $\to$ (c) is clear.

(c) $\to$ (a). Let $A \in \overline{\mathcal{L}}$ and $T \nvdash_c A$. Then there is a model $\mathcal{M}_0$ of $T \cup \{\neg A\}$. Now let $\mathcal{M} \in \mathrm{Mod}_{\mathcal{L}}(T)$ be arbitrary. By (c) we have $\mathcal{M} \equiv \mathcal{M}_0$, hence $\mathcal{M} \models \neg A$. Therefore $T \vdash_c \neg A$. □

2.2.4. Elementary equivalence and isomorphism.

LEMMA. *Let π be an isomorphism between $\mathcal{M}$ and $\mathcal{M}'$. Then for all terms t and formulas A and for every sufficiently big assignment η in $|\mathcal{M}|$*

(a) *$\pi(t^{\mathcal{M}}[\eta]) =^{\mathcal{M}'} t^{\mathcal{M}'}[\pi \circ \eta]$ and*
(b) *$\mathcal{M} \models A[\eta] \leftrightarrow \mathcal{M}' \models A[\pi \circ \eta]$. In particular,*

$$\mathcal{M} \cong \mathcal{M}' \to \mathcal{M} \equiv \mathcal{M}'.$$

PROOF. (a). Induction on t. For simplicity we only consider the case of a unary function symbol.

$$\begin{aligned}\pi(x^{\mathcal{M}}[\eta]) &= \pi(\eta(x)) = x^{\mathcal{M}'}[\pi \circ \eta]\\ \pi((ft)^{\mathcal{M}}[\eta]) &= \pi(f^{\mathcal{M}}(t^{\mathcal{M}}[\eta]))\\ &=^{\mathcal{M}'} f^{\mathcal{M}'}(\pi(t^{\mathcal{M}}[\eta]))\\ &=^{\mathcal{M}'} f^{\mathcal{M}'}(t^{\mathcal{M}'}[\pi \circ \eta])\\ &= (ft)^{\mathcal{M}'}[\pi \circ \eta].\end{aligned}$$

(b). Induction on A. For simplicity we only consider the case of a unary relation symbol P and the case $\forall_x A$.

$$\begin{aligned}\mathcal{M} \models (Pr)[\eta] &\leftrightarrow P^{\mathcal{M}}(r^{\mathcal{M}}[\eta])\\ &\leftrightarrow P^{\mathcal{M}'}(\pi(r^{\mathcal{M}}[\eta]))\\ &\leftrightarrow P^{\mathcal{M}'}(r^{\mathcal{M}'}[\pi \circ \eta])\\ &\leftrightarrow \mathcal{M}' \models (Pr)[\pi \circ \eta],\end{aligned}$$

$$\begin{aligned}\mathcal{M} \models \forall_x A[\eta] &\leftrightarrow \forall_{a \in |\mathcal{M}|}(\mathcal{M} \models A[\eta^a_x])\\ &\leftrightarrow \forall_{a \in |\mathcal{M}|}(\mathcal{M}' \models A[\pi \circ \eta^a_x])\\ &\leftrightarrow \forall_{a \in |\mathcal{M}|}(\mathcal{M}' \models A[(\pi \circ \eta)^{\pi(a)}_x])\\ &\leftrightarrow \forall_{a' \in |\mathcal{M}'|}(\mathcal{M}' \models A[(\pi \circ \eta)^{a'}_x])\\ &\leftrightarrow \mathcal{M}' \models \forall_x A[\pi \circ \eta].\end{aligned}$$

For part "$\rightarrow$" of the next-to-last equivalence we have used the Coincidence Lemma from 2.2. □

The converse, i.e., that $\mathcal{M} \equiv \mathcal{M}'$ implies $\mathcal{M} \cong \mathcal{M}'$, is true for finite models, but not for infinite ones. This proves the impossibility to characterize models by first order axioms.

THEOREM. *For every infinite model $\mathcal{M}$ there is an elementarily equivalent model $\mathcal{M}_0$ not isomorphic to $\mathcal{M}$.*

PROOF. Let $=^{\mathcal{M}}$ be the equality on $D := |\mathcal{M}|$, and let $\mathcal{P}(D)$ denote the power set of D. For every $\alpha \in \mathcal{P}(D)$ choose a new constant c_α. In the language $\mathcal{L}' := \mathcal{L} \cup \{\, c_\alpha \mid \alpha \in \mathcal{P}(D) \,\}$ we consider the axiom system

$$\Gamma := \mathrm{Th}(\mathcal{M}) \cup \{\, c_\alpha \neq c_\beta \mid \alpha, \beta \in \mathcal{P}(D) \text{ and } \alpha \neq \beta \,\} \cup \mathrm{Eq}_{\mathcal{L}'}.$$

Every finite subset of Γ is satisfiable by an appropriate expansion of $\mathcal{M}$. Hence by the general compactness theorem also Γ is satisfiable, say by $\mathcal{M}'_0$. Let $\mathcal{M}_0 := \mathcal{M}'_0 \restriction \mathcal{L}$. We may assume that $=^{\mathcal{M}_0}$ is the equality on $|\mathcal{M}_0|$. $\mathcal{M}_0$

is not isomorphic to $\mathcal{M}$, for otherwise we would have an injection of $\mathcal{P}(D)$ into D and therefore a contradiction. □

2.3. Applications

2.3.1. Non-standard models. By what we just proved it is impossible to characterize an infinite model by a first order axiom system up to isomorphism. However, if we extend first order logic by also allowing quantification over sets X, we can formulate the following *Peano axioms*

$$\forall_n(\mathrm{S}n \neq 0),$$
$$\forall_{n,m}(\mathrm{S}n = \mathrm{S}m \to n = m),$$
$$\forall_X(0 \in X \to \forall_n(n \in X \to \mathrm{S}n \in X) \to \forall_n(n \in X)).$$

One can show easily that $(\mathbb{N}, 0, \mathrm{S})$ is up to isomorphism the unique model of the Peano axioms. A model which is elementarily equivalent, but not isomorphic to $\mathcal{N} := (\mathbb{N}, 0, \mathrm{S})$, is called a *non-standard model* of $\mathcal{N}$. In such non-standard models the principle of complete induction does not hold for all subsets of $|\mathcal{N}|$.

THEOREM. *There are countable non-standard models of the natural numbers.*

PROOF. Let x be a variable and $\Gamma := \mathrm{Th}(\mathcal{N}) \cup \{\, x \neq \underline{n} \mid n \in \mathbb{N} \,\}$, where $\underline{0} := 0$ and $\underline{n+1} := \mathrm{S}\underline{n}$. Clearly every finite subset of Γ is satisfiable, hence by compactness also Γ. By the theorem of Löwenheim and Skolem we then have a countable or finite $\mathcal{M}$ and an assignment η such that $\mathcal{M} \models \Gamma[\eta]$. Because of $\mathcal{M} \models \mathrm{Th}(\mathcal{N})$ we have $\mathcal{M} \equiv \mathcal{N}$ by 2.2.3; hence $\mathcal{M}$ is countable. Moreover $\eta(x) \neq^{\mathcal{M}} \underline{n}^{\mathcal{M}}$ for all $n \in \mathbb{N}$, hence $\mathcal{M} \ncong \mathcal{N}$. □

2.3.2. Archimedian ordered fields. We now consider some easy applications to well-known axiom systems. The axioms of *field theory* are (the equality axioms and)

$$x + (y + z) = (x + y) + z, \qquad x \cdot (y \cdot z) = (x \cdot y) \cdot z,$$
$$0 + x = x, \qquad 1 \cdot x = x,$$
$$(-x) + x = 0, \qquad x \neq 0 \to x^{-1} \cdot x = 1,$$
$$x + y = y + x, \qquad x \cdot y = y \cdot x,$$

and also

$$(x + y) \cdot z = (x \cdot z) + (y \cdot z),$$
$$1 \neq 0.$$

Fields are the models of this axiom system.

In the theory of ordered fields one has in addition a binary relation symbol $<$ and as axioms

$$\begin{aligned}
&x \not< x,\\
&x < y \to y < z \to x < z,\\
&x < y \vee x = y \vee y < x,\\
&x < y \to x + z < y + z,\\
&0 < x \to 0 < y \to 0 < x \cdot y.
\end{aligned}$$

Ordered fields are the models of this extended axiom system. An ordered field is called *archimedian ordered* if for every element a of the field there is a natural number n such that a is less than the n-fold multiple of the 1 in the field.

THEOREM. *For every archimedian ordered field there is an elementarily equivalent ordered field that is not archimedian ordered.*

PROOF. Let $\mathcal{K}$ be an archimedian ordered field, x a variable and

$$\Gamma := \mathrm{Th}(\mathcal{K}) \cup \{\, \underline{n} < x \mid n \in \mathbb{N} \,\}.$$

Clearly every finite subset of Γ is satisfiable, hence by the general compactness theorem also Γ. Therefore we have $\mathcal{M}$ and η such that $\mathcal{M} \models \Gamma[\eta]$. Because of $\mathcal{M} \models \mathrm{Th}(\mathcal{K})$ we obtain $\mathcal{M} \equiv \mathcal{K}$ and hence $\mathcal{M}$ is an ordered field. Moreover $1^{\mathcal{M}} \cdot n <^{\mathcal{M}} \eta(x)$ for all $n \in \mathbb{N}$, hence $\mathcal{M}$ is not archimedian ordered. □

2.3.3. Axiomatizable models. A class $\mathcal{S}$ of $\mathcal{L}$-models is (*finitely*) *axiomatizable* if there is a (finite) axiom system Γ such that $\mathcal{S} = \mathrm{Mod}_{\mathcal{L}}(\Gamma)$. Clearly $\mathcal{S}$ is finitely axiomatizable if and only if $\mathcal{S} = \mathrm{Mod}_{\mathcal{L}}(\{A\})$ for some formula A. If for every $\mathcal{M} \in \mathcal{S}$ there is an elementarily equivalent $\mathcal{M}' \notin \mathcal{S}$, then $\mathcal{S}$ cannot possibly be axiomatizable. By the theorem above we can conclude that the class of archimedian ordered fields is not axiomatizable. It also follows that the class of non archimedian ordered fields is not axiomatizable.

LEMMA. *Let $\mathcal{S}$ be a class of $\mathcal{L}$-models and Γ an axiom system.*

(a) *$\mathcal{S}$ is finitely axiomatizable if and only if $\mathcal{S}$ and the complement of $\mathcal{S}$ are axiomatizable.*
(b) *If $\mathrm{Mod}_{\mathcal{L}}(\Gamma)$ is finitely axiomatizable, then there is a finite $\Gamma_0 \subseteq \Gamma$ such that $\mathrm{Mod}_{\mathcal{L}}(\Gamma_0) = \mathrm{Mod}_{\mathcal{L}}(\Gamma)$.*

PROOF. (a). Let $\mathcal{S}^{\mathrm{C}}$ denote the complement of $\mathcal{S}$. For $\to$ assume $\mathcal{S} = \mathrm{Mod}_{\mathcal{L}}(\{A\})$. Then $\mathcal{M} \in \mathcal{S}^{\mathrm{C}} \leftrightarrow \mathcal{M} \models \neg A$, hence $\mathcal{S}^{\mathrm{C}} = \mathrm{Mod}_{\mathcal{L}}(\{\neg A\})$.

For the converse. assume $\mathcal{S} = \mathrm{Mod}_{\mathcal{L}}(\Gamma_1)$ and $\mathcal{S}^{\mathrm{C}} = \mathrm{Mod}_{\mathcal{L}}(\Gamma_2)$. Then $\Gamma_1 \cup \Gamma_2$ is not satisfiable, hence there is a finite $\Gamma \subseteq \Gamma_1$ such that $\Gamma \cup \Gamma_2$ is not satisfiable. One obtains

$$\mathcal{M} \in \mathcal{S} \to \mathcal{M} \models \Gamma \to \mathcal{M} \not\models \Gamma_2 \to \mathcal{M} \notin \mathcal{S}^{\mathrm{C}} \to \mathcal{M} \in \mathcal{S}.$$

Hence $\mathcal{S} = \mathrm{Mod}_{\mathcal{L}}(\Gamma)$.

(b). Let $\mathrm{Mod}_{\mathcal{L}}(\Gamma) = \mathrm{Mod}_{\mathcal{L}}(\{A\})$. Then $\Gamma \vdash_c A$, hence also $\Gamma_0 \vdash_c A$ for a finite $\Gamma_0 \subseteq \Gamma$. One obtains

$$\mathcal{M} \models \Gamma \to \mathcal{M} \models \Gamma_0 \to \mathcal{M} \models A \to \mathcal{M} \models \Gamma.$$

Hence $\mathrm{Mod}_{\mathcal{L}}(\Gamma_0) = \mathrm{Mod}_{\mathcal{L}}(\Gamma)$. □

2.3.4. Dense linear orders without end points. Finally we consider as an example of a complete theory the theory DO of dense linear orders without end points. The axioms are (the equality axioms and)

$$\begin{aligned} &x \not< x, && x < y \to \exists_z(x < z \wedge z < y),\\ &x < y \to y < z \to x < z, && \exists_y(x < y),\\ &x < y \vee x = y \vee y < x, && \exists_y(y < x). \end{aligned}$$

Lemma. *Every countable model of* DO *is isomorphic to the model* $(\mathbb{Q}, <)$ *of rational numbers.*

Proof. Let $\mathcal{M} = (D, \prec)$ be a countable model of DO; we can assume that $=^{\mathcal{M}}$ is the equality on D. Let $D = \{\, b_n \mid n \in \mathbb{N} \,\}$ and $\mathbb{Q} = \{\, a_n \mid n \in \mathbb{N} \,\}$, where we may assume $a_n \neq a_m$ and $b_n \neq b_m$ for $n < m$. We define recursively functions $f_n \subseteq \mathbb{Q} \times D$ as follows. Let $f_0 := \{(a_0, b_0)\}$. Assume we have already constructed f_n.

Case $n+1 = 2m$. Let j be minimal such that $b_j \notin \mathrm{ran}(f_n)$. Choose $a_i \notin \mathrm{dom}(f_n)$ such that for all $a \in \mathrm{dom}(f_n)$ we have $a_i < a \leftrightarrow b_j < f_n(a)$; such an a_i exists, since $\mathcal{M}$ and $(\mathbb{Q}, <)$ are models of DO. Let $f_{n+1} := f_n \cup \{(a_i, b_j)\}$.

Case $n + 1 = 2m + 1$. This is treated similarly. Let i be minimal such that $a_i \notin \mathrm{dom}(f_n)$. Choose $b_j \notin \mathrm{ran}(f_n)$ such that for all $a \in \mathrm{dom}(f_n)$ we have $a_i < a \leftrightarrow b_j < f_n(a)$; such a b_j exists, since $\mathcal{M}$ and $(\mathbb{Q}, <)$ are models of DO. Let $f_{n+1} := f_n \cup \{(a_i, b_j)\}$.

Then $\{b_0, \ldots, b_m\} \subseteq \mathrm{ran}(f_{2m})$ and $\{a_0, \ldots, a_{m+1}\} \subseteq \mathrm{dom}(f_{2m+1})$ by construction, and $f := \bigcup_n f_n$ is an isomorphism of $(\mathbb{Q}, <)$ onto $\mathcal{M}$. □

Theorem. *The theory* DO *is complete, and* $\mathrm{DO} = \mathrm{Th}(\mathbb{Q}, <)$.

Proof. Clearly $(\mathbb{Q}, <)$ is a model of DO. Hence by 2.2.3 it suffices to show that for every model $\mathcal{M}$ of DO we have $\mathcal{M} \equiv (\mathbb{Q}, <)$. So let $\mathcal{M}$ model of DO. By 2.2.3 there is a countable $\mathcal{M}'$ such that $\mathcal{M} \equiv \mathcal{M}'$. By the preceding lemma $\mathcal{M}' \cong (\mathbb{Q}, <)$, hence $\mathcal{M} \equiv \mathcal{M}' \equiv (\mathbb{Q}, <)$. □

A further example of a complete theory is the theory of algebraically closed fields. For a proof of this fact and for many more subjects of model theory we refer to the literature (e.g., Chang and Keisler (1990) or Ebbinghaus et al. (1996)).

CHAPTER 3

Recursion Theory

In this chapter we develop the basics of recursive function theory, or as it is more generally known, computability theory. Its history goes back to the seminal works of Turing, Kleene and others in the 1930's.

A computable function is one defined by a program whose operational semantics tell an idealized computer what to do to its storage locations as it proceeds deterministically from input to output, without any prior restrictions on storage space or computation time. We shall be concerned with various program-styles and the relationships between them, but the emphasis throughout will be on one underlying data-type, namely the natural numbers, since it is there that the most basic foundational connections between proof theory and computation are to be seen in their clearest light.

The two best-known models of machine computation are the Turing Machine and the (Unlimited) Register Machine of Shepherdson and Sturgis (1963). We base our development on the latter since it affords the quickest route to the results we want to establish.

3.1. Register Machines

3.1.1. Programs. A *register machine* stores natural numbers in registers denoted u, v, w, x, y, z possibly with subscripts, and it responds step by step to a *program* consisting of an ordered list of basic instructions:

$$\begin{array}{l} I_0 \\ I_1 \\ \vdots \\ I_{k-1} \end{array}$$

Each instruction has one of the following three forms whose meanings are obvious:

Zero: $x := 0$,
Succ: $x := x + 1$,
Jump: [**if** $x = y$ **then** I_n **else** I_m].

The instructions are obeyed in order starting with I_0 except when a conditional jump instruction is encountered, in which case the next instruction

will be either I_n or I_m according as the numerical contents of registers x and y are equal or not at that stage. The computation *terminates* when it runs out of instructions, that is when the next instruction called for is I_k. Thus if a program of length k contains a jump instruction as above then it must satisfy the condition $n, m \leq k$ and I_k means "halt". Notice of course that some programs do not terminate, for example the following one-liner:

$$[\textbf{if } x = x \textbf{ then } I_0 \textbf{ else } I_1]$$

3.1.2. Program constructs. We develop some shorthand for building up standard sorts of programs.

Transfer. "$x := y$" is the program

$$\begin{array}{l} x := 0 \\ [\textbf{if } x = y \textbf{ then } I_4 \textbf{ else } I_2] \\ x := x + 1 \\ [\textbf{if } x = x \textbf{ then } I_1 \textbf{ else } I_1], \end{array}$$

which copies the contents of register y into register x.

Predecessor. The program "$x := y \dot{-} 1$" copies the modified predecessor of y into x, and simultaneously copies y into z:

$$\begin{array}{l} x := 0 \\ z := 0 \\ [\textbf{if } x = y \textbf{ then } I_8 \textbf{ else } I_3] \\ z := z + 1 \\ [\textbf{if } z = y \textbf{ then } I_8 \textbf{ else } I_5] \\ z := z + 1 \\ x := x + 1 \\ [\textbf{if } z = y \textbf{ then } I_8 \textbf{ else } I_5]. \end{array}$$

Composition. "P **;** Q" is the program obtained by concatenating program P with program Q. However in order to ensure that jump instructions in Q of the form "$[\textbf{if } x = y \textbf{ then } I_n \textbf{ else } I_m]$" still operate properly within Q they need to be re-numbered by changing the addresses n, m to $k+n, k+m$ respectively where k is the length of program P. Thus the effect of this program is to do P until it halts (if ever) and then do Q.

Conditional. "**if** $x = y$ **then** P **else** Q **fi**" is the program

$$\begin{array}{l} [\textbf{if } x = y \textbf{ then } I_1 \textbf{ else } I_{k+2}] \\ \vdots P \\ [\textbf{if } x = x \textbf{ then } I_{k+2+l} \textbf{ else } I_2] \\ \vdots Q \end{array}$$

where k, l are the lengths of the programs P, Q respectively, and again their jump instructions must be appropriately renumbered by adding 1 to the

addresses in P and $k+2$ to the addresses in Q. Clearly if $x = y$ then program P is obeyed and the next jump instruction automatically bypasses Q and halts. If $x \neq y$ then program Q is performed.

For Loop. "**for** $i = 1 \dots x$ **do** P **od**" is the program

$$\begin{array}{l} i := 0 \\ [\textbf{if } x = i \textbf{ then } I_{k+4} \textbf{ else } I_2] \\ i := i + 1 \\ \vdots P \\ [\textbf{if } x = i \textbf{ then } I_{k+4} \textbf{ else } I_2] \end{array}$$

where again, k is the length of program P and the jump instructions in P must be appropriately re-addressed by adding 3. The intention of this new program is that it should iterate the program P x times (do nothing if $x = 0$). This requires the restriction that the register x and the "local" counting-register i are not re-assigned new values inside P.

While Loop. "**while** $x \neq 0$ **do** P **od**" is the program

$$\begin{array}{l} y := 0 \\ [\textbf{if } x = y \textbf{ then } I_{k+3} \textbf{ else } I_2] \\ \vdots P \\ [\textbf{if } x = y \textbf{ then } I_{k+3} \textbf{ else } I_2] \end{array}$$

where again, k is the length of program P and the jump instructions in P must be re-addressed by adding 2. This program keeps on doing P until (if ever) the register x becomes 0; it requires the restriction that the auxiliary register y is not re-assigned new values inside P.

3.1.3. Register machine computable functions. A register machine program P may have certain distinguished "input registers" and "output registers". It may also use other "working registers" for scratchwork and these will initially be set to zero. We write $P(x_1, \dots, x_k; y)$ to signify that program P has input registers $x_1, \dots, x_k$ and one output register y, which are distinct.

DEFINITION. The program $P(x_1, \dots, x_k; y)$ is said to *compute* the k-ary partial function $\varphi\colon \mathbb{N}^k \to \mathbb{N}$ if, starting with any numerical values $n_1, \dots, n_k$ in the input registers, the program terminates with the number m in the output register if and only if $\varphi(n_1, \dots, n_k)$ is defined with value m. In this case, the input registers hold their original values.

A function is *register machine computable* if there is some program which computes it.

Here are some examples.

Addition. "Add$(x, y; z)$" is the program

$$z := x \ \textbf{;}\ \textbf{for}\ i = 1, \ldots, y\ \textbf{do}\ z := z + 1\ \textbf{od}$$

which adds the contents of registers x and y into register z.

Subtraction. "Subt$(x, y; z)$" is the program

$$z := x \ \textbf{;}\ \textbf{for}\ i = 1, \ldots, y\ \textbf{do}\ w := z \dot{-} 1 \ \textbf{;}\ z := w\ \textbf{od}$$

which computes the modified subtraction function $x \dot{-} y$.

Bounded Sum. If $P(x_1, \ldots, x_k, w; y)$ computes the $k+1$-ary function φ then the program $Q(x_1, \ldots, x_k, z; x)$:

$x := 0$ **;**
for $i = 1, \ldots, z$ **do** $w := i \dot{-} 1$ **;** $P(\vec{x}, w; y)$ **;** $v := x$ **;** Add$(v, y; x)$ **od**

computes the function

$$\psi(x_1, \ldots, x_k, z) = \sum_{w<z} \varphi(x_1, \ldots, x_k, w)$$

which will be undefined if for some $w < z$, $\varphi(x_1, \ldots, x_k, w)$ is undefined.

Multiplication. Deleting "$w := i \dot{-} 1$ **;** P" from the last example gives a program Mult$(z, y; x)$ which places the product of y and z into x.

Bounded Product. If in the bounded sum example, the instruction $x := x + 1$ is inserted immediately after $x := 0$, and if Add$(v, y; x)$ is replaced by Mult$(v, y; x)$, then the resulting program computes the function

$$\psi(x_1, \ldots, x_k, z) = \prod_{w<z} \varphi(x_1, \ldots, x_k, w).$$

Composition. If $P_j(x_1, \ldots, x_k; y_j)$ computes φ_j for each $j = i, \ldots, n$ and if $P_0(y_1, \ldots, y_n; y_0)$ computes φ_0, then the program $Q(x_1, \ldots, x_k; y_0)$:

$$P_1(x_1, \ldots, x_k; y_1)\ \textbf{;}\ \ldots\ \textbf{;}\ P_n(x_1, \ldots, x_k; y_n)\ \textbf{;}\ P_0(y_1, \ldots, y_n; y_0)$$

computes the function

$$\psi(x_1, \ldots, x_k) = \varphi_0(\varphi_1(x_1, \ldots, x_k), \ldots, \varphi_n(x_1, \ldots, x_k))$$

which will be undefined if any of the φ-subterms on the right hand side is undefined.

Unbounded Minimization. If $P(x_1, \ldots, x_k, y; z)$ computes φ then the program $Q(x_1, \ldots, x_k; z)$:

$y := 0$ **;** $z := 0$ **;** $z := z + 1$ **;**
while $z \neq 0$ **do** $P(x_1, \ldots, x_k, y; z)$ **;** $y := y + 1$ **od ;**
$z := y \dot{-} 1$

computes the function

$$\psi(x_1, \ldots, x_k) \ = \ \mu_y(\varphi(x_1, \ldots, x_k, y) = 0)$$

that is, the *least number* y such that $\varphi(x_1, \ldots, x_k, y')$ is defined for every $y' \leq y$ and $\varphi(x_1, \ldots, x_k, y) = 0$.

3.2. Elementary Functions

3.2.1. Definition and simple properties. The *elementary functions* of Kalmár (1943) are those number-theoretic functions which can be defined explicitly by compositional terms built up from variables and the constants $0, 1$ by repeated applications of addition $+$, modified subtraction $\dot{-}$, bounded sums and bounded products.

By omitting bounded products, one obtains the *subelementary* functions.

The examples in the previous section show that all elementary functions are computable and totally defined. Multiplication and exponentiation are elementary since

$$m \cdot n = \textstyle\sum_{i<n} m \text{ and } m^n = \prod_{i<n} m$$

and hence by repeated composition, all exponential polynomials are elementary.

In addition the elementary functions are closed under

Definition by Cases.

$$f(\vec{n}\,) = \begin{cases} g_0(\vec{n}\,) & \text{if } h(\vec{n}\,) = 0 \\ g_1(\vec{n}\,) & \text{otherwise} \end{cases}$$

since f can be defined from g_0, g_1 and h by

$$f(\vec{n}\,) = g_0(\vec{n}\,) \cdot (1 \dot{-} h(\vec{n}\,)) + g_1(\vec{n}\,) \cdot (1 \dot{-} (1 \dot{-} h(\vec{n}\,))).$$

Bounded Minimization.

$$f(\vec{n}, m) = \mu_{k<m}(g(\vec{n}, k) = 0)$$

since f can be defined from g by

$$f(\vec{n}, m) = \sum_{i<m} \big(1 \dot{-} \sum_{k \leq i} (1 \dot{-} g(\vec{n}, k))\big).$$

Note: this definition gives value m if there is no $k < m$ such that $g(\vec{n}, k) = 0$. It shows that not only the elementary, but in fact the subelementary functions are closed under bounded minimization. Furthermore, we define $\mu_{k \leq m}(g(\vec{n}, k) = 0)$ as $\mu_{k<m+1}(g(\vec{n}, k) = 0)$.

Lemma.

(a) *For every elementary function* $f\colon \mathbb{N}^r \to \mathbb{N}$ *there is a number* k *such that for all* $\vec{n} = n_1, \ldots, n_r$,

$$f(\vec{n}\,) < 2_k(\max(\vec{n}\,))$$

where $2_0(m) := m$ *and* $2_{k+1}(m) := 2^{2_k(m)}$.

(b) *Hence the function $n \mapsto 2_n(1)$ is not elementary.*

PROOF. (a). By induction on the build-up of the compositional term defining f. The result clearly holds if f is any one of the base functions:

$$f(\vec{n}\,) = 0 \text{ or } 1 \text{ or } n_i \text{ or } n_i + n_j \text{ or } n_i \dot{-} n_j.$$

If f is defined from g by application of bounded sum or product:

$$f(\vec{n}, m) = \sum_{i<m} g(\vec{n}, i) \text{ or } \prod_{i<m} g(\vec{n}, i)$$

where $g(\vec{n}, i) < 2_k(\max(\vec{n}, i))$ then we have

$$f(\vec{n}, m) \leq (2_k(\max(\vec{n}, m)))^m < 2_{k+2}(\max(\vec{n}, m))$$

using $n^n < 2^{2^n}$ (since $n^n < (2^n)^n \leq 2^{2^n}$ for $n \geq 3$).

If f is defined from $g_0, g_1, \ldots, g_l$ by composition:

$$f(\vec{n}\,) = g_0(g_1(\vec{n}\,), \ldots, g_l(\vec{n}\,))$$

where for each $j \leq l$ we have $g_j(-) < 2_{k_j}(\max(-))$, then with $k = \max_j k_j$,

$$f(\vec{n}\,) < 2_k(2_k(\max(\vec{n}\,))) = 2_{2k}(\max(\vec{n}\,))$$

and this completes the first part.

(b). If $2_n(1)$ were an elementary function of n then by (a) there would be a positive k such that for all n,

$$2_n(1) < 2_k(n)$$

but then putting $n = 2_k(1)$ yields $2_{2_k(1)}(1) < 2_{2k}(1)$, a contradiction. □

3.2.2. Elementary relations. A relation R on $\mathbb{N}^k$ is said to be *elementary* if its characteristic function

$$c_R(\vec{n}\,) = \begin{cases} 1 & \text{if } R(\vec{n}\,) \\ 0 & \text{otherwise} \end{cases}$$

is elementary. In particular, the "equality" and "less than" relations are elementary since their characteristic functions can be defined as follows:

$$c_<(n, m) = 1 \dot{-} (1 \dot{-} (m \dot{-} n)), \quad c_=(n, m) = 1 \dot{-} (c_<(n, m) + c_<(m, n)).$$

Furthermore if R is elementary then so is the function

$$f(\vec{n}, m) = \mu_{k<m} R(\vec{n}, k)$$

since $R(\vec{n}, k)$ is equivalent to $1 \dot{-} c_R(\vec{n}, k) = 0$.

LEMMA. *The elementary relations are closed under applications of propositional connectives and bounded quantifiers.*

PROOF. For example, the characteristic function of $\neg R$ is

$$1 \dot{-} c_R(\vec{n}\,).$$

The characteristic function of $R_0 \wedge R_1$ is

$$c_{R_0}(\vec{n}\,) \cdot c_{R_1}(\vec{n}\,).$$

The characteristic function of $\forall_{i<m} R(\vec{n}, i)$ is

$$c_=(m, \mu_{i<m}(c_R(\vec{n}, i) = 0)). \qquad \square$$

EXAMPLES. The above closure properties enable us to show that many "natural" functions and relations of number theory are elementary; thus

$$\begin{aligned}
\lfloor \frac{n}{m} \rfloor \quad &= \mu_{k<n}(n < (k+1)m), \\
n \bmod m &= n \dot{-} \lfloor \frac{n}{m} \rfloor m, \\
\mathrm{Prime}(n) &\leftrightarrow 1 < n \wedge \neg \exists_{m<n}(1 < m \wedge n \bmod m = 0), \\
p_n \quad &= \mu_{m<2^{2^n}}(\mathrm{Prime}(m) \wedge n = \sum_{i<m} c_{\mathrm{Prime}}(i)),
\end{aligned}$$

so $p_0, p_1, p_2, \ldots$ gives the enumeration of primes in increasing order. The estimate $p_n \leq 2^{2^n}$ for the nth prime p_n can be proved by induction on n: For $n = 0$ this is clear, and for $n \geq 1$ we obtain

$$p_n \leq p_0 p_1 \cdots p_{n-1} + 1 \leq 2^{2^0} 2^{2^1} \cdots 2^{2^{n-1}} + 1 = 2^{2^n - 1} + 1 < 2^{2^n}.$$

3.2.3. The class $\mathcal{E}$.

DEFINITION. The class $\mathcal{E}$ consists of those number theoretic functions which can be defined from the initial functions: constant 0, successor S, projections (onto the ith coordinate), addition $+$, modified subtraction $\dot{-}$, multiplication $\cdot$ and exponentiation 2^x, by applications of composition and bounded minimization.

The remarks above show immediately that the characteristic functions of the equality and less than relations lie in $\mathcal{E}$, and that (by the proof of the lemma) the relations in $\mathcal{E}$ are closed under propositional connectives and bounded quantifiers.

Furthermore the above examples show that all the functions in the class $\mathcal{E}$ are elementary. We now prove the converse, which will be useful later.

LEMMA. *There are "pairing functions" π, π_1, π_2 in $\mathcal{E}$ with the following properties:*

(a) π *maps* $\mathbb{N} \times \mathbb{N}$ *bijectively onto* $\mathbb{N}$,
(b) $\pi(a, b) + b + 2 \leq (a + b + 1)^2$ *for* $a + b \geq 1$, *hence* $\pi(a, b) < (a + b + 1)^2$,
(c) $\pi_1(c), \pi_2(c) \leq c$,

(d) $\pi(\pi_1(c), \pi_2(c)) = c$,
(e) $\pi_1(\pi(a, b)) = a$,
(f) $\pi_2(\pi(a, b)) = b$.

PROOF. Enumerate the pairs of natural numbers as follows:

$$\begin{array}{cccccc} \vdots & & & & & \\ 6 & \dots & & & & \\ 3 & 7 & \dots & & & \\ 1 & 4 & 8 & \dots & & \\ 0 & 2 & 5 & 9 & \dots & \end{array}$$

At position $(0, b)$ we clearly have the sum of the lengths of the preceding diagonals, and on the next diagonal $a + b$ remains constant. Let $\pi(a, b)$ be the number written at position (a, b). Then we have

$$\pi(a, b) = \big(\sum_{i \le a+b} i\big) + a = \frac{1}{2}(a + b)(a + b + 1) + a.$$

Clearly $\pi\colon \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ is bijective. Moreover, $a, b \le \pi(a, b)$ and in case $\pi(a, b) \neq 0$ also $a < \pi(a, b)$. Let

$$\pi_1(c) := \mu_{x \le c} \exists_{y \le c} (\pi(x, y) = c),$$
$$\pi_2(c) := \mu_{y \le c} \exists_{x \le c} (\pi(x, y) = c).$$

Then clearly $\pi_i(c) \le c$ for $i \in \{1, 2\}$ and

$$\pi_1(\pi(a, b)) = a, \quad \pi_2(\pi(a, b)) = b, \quad \pi(\pi_1(c), \pi_2(c)) = c.$$

π, π_1 and π_2 are in $\mathcal{E}$ by definition. For $\pi(a, b)$ we have the estimate

$$\pi(a, b) + b + 2 \le (a + b + 1)^2 \quad \text{for } a + b \ge 1.$$

This follows with $n := a + b$ from

$$\frac{1}{2} n(n + 1) + n + 2 \le (n + 1)^2 \quad \text{for } n \ge 1,$$

which is equivalent to $n(n + 1) + 2(n + 1) \le 2((n + 1)^2 - 1)$ and hence to $(n + 2)(n + 1) \le 2n(n + 2)$, which holds for $n \ge 1$. □

The proof shows that π, π_1 and π_2 are in fact subelementary.

THEOREM (Gödel's β-function). *There is in $\mathcal{E}$ a function β with the following property: For every sequence $a_0, \dots, a_{n-1} < b$ of numbers less than b we can find a number $c \le 4 \cdot 4^{n(b+n+1)^4}$ such that $\beta(c, i) = a_i$ for all $i < n$.*

PROOF. Let

$$a := \pi(b,n) \quad \text{and} \quad d := \prod_{i<n}\big(1+\pi(a_i,i)a!\big).$$

From $a!$ and d we can, for each given $i < n$, reconstruct the number a_i as the unique $x < b$ such that

$$1+\pi(x,i)a! \mid d.$$

For clearly a_i is such an x, and if some $x < b$ were to satisfy the same condition, then because $\pi(x,i) < a$ and the numbers $1+ka!$ are relatively prime for $k \le a$, we would have $\pi(x,i) = \pi(a_j,j)$ for some $j < n$. Hence $x = a_j$ and $i = j$, thus $x = a_i$. – Therefore

$$a_i = \mu_{x<b}{}^{\exists}{}_{z<d}((1+\pi(x,i)a!)z = d).$$

We can now define Gödel's β-function as

$$\beta(c,i) := \mu_{x<\pi_1(c)}{}^{\exists}{}_{z<\pi_2(c)}((1+\pi(x,i)\cdot\pi_1(c))\cdot z = \pi_2(c)).$$

Clearly β is in $\mathcal{E}$. Furthermore with $c := \pi(a!,d)$ we see that $\beta(c,i) = a_i$. It is then not difficult to estimate the given bound on c, using $\pi(b,n) < (b+n+1)^2$. □

The above definition of β shows that it is subelementary.

3.2.4. Closure properties of $\mathcal{E}$.

THEOREM. *The class $\mathcal{E}$ is closed under limited recursion. Thus if g, h, k are given functions in $\mathcal{E}$ and f is defined from them according to the schema*

$$\begin{aligned} f(\vec{m},0) &= g(\vec{m}),\\ f(\vec{m},n+1) &= h(n,f(\vec{m},n),\vec{m}),\\ f(\vec{m},n) &\le k(\vec{m},n), \end{aligned}$$

then f is in $\mathcal{E}$ also.

PROOF. Let f be defined from g, h and k in $\mathcal{E}$, by limited recursion as above. Using Gödel's β-function as in the last theorem we can find for any given $\vec{m}, n$ a number c such that $\beta(c,i) = f(\vec{m},i)$ for all $i \le n$. Let $R(\vec{m},n,c)$ be the relation

$$\beta(c,0) = g(\vec{m}) \wedge \forall_{i<n}(\beta(c,i+1) = h(i,\beta(c,i),\vec{m}))$$

and note by the remarks above that its characteristic function is in $\mathcal{E}$. It is clear, by induction, that if $R(\vec{m},n,c)$ holds then $\beta(c,i) = f(\vec{m},i)$, for all $i \le n$. Therefore we can define f explicitly by the equation

$$f(\vec{m},n) = \beta(\mu_c R(\vec{m},n,c),n).$$

f will lie in $\mathcal{E}$ if μ_c can be bounded by an $\mathcal{E}$ function. However, the theorem on Gödel's β-function gives a bound $4 \cdot 4^{(n+1)(b+n+2)^4}$, where in this case b can be taken as the maximum of $k(\vec{m}, i)$ for $i \leq n$. But this can be defined in $\mathcal{E}$ as $k(\vec{m}, i_0)$, where $i_0 = \mu_{i \leq n} \forall_{j \leq n}(k(\vec{m}, j) \leq k(\vec{m}, i))$. Hence μ_c can be bounded by an $\mathcal{E}$ function. □

REMARK. Note that it is in this proof only that the exponential function is required, in providing a bound for μ.

COROLLARY. *$\mathcal{E}$ is the class of all elementary functions.*

PROOF. It is sufficient merely to show that $\mathcal{E}$ is closed under bounded sums and bounded products. Suppose for instance, that f is defined from g in $\mathcal{E}$ by bounded summation: $f(\vec{m}, n) = \sum_{i<n} g(\vec{m}, i)$. Then f can be defined by limited recursion, as follows

$$\begin{aligned} f(\vec{m}, 0) &= 0 \\ f(\vec{m}, n+1) &= f(\vec{m}, n) + g(\vec{m}, n) \\ f(\vec{m}, n) &\leq n \cdot \max_{i<n} g(\vec{m}, i) \end{aligned}$$

and the functions (including the bound) from which it is defined are in $\mathcal{E}$. Thus f is in $\mathcal{E}$ by the theorem. If instead, f is defined by bounded product, then proceed similarly. □

3.2.5. Coding finite lists. Computation on lists is a practical necessity, so because we are basing everything here on the single data type $\mathbb{N}$ we must develop some means of "coding" finite lists or sequences of natural numbers into $\mathbb{N}$ itself. There are various ways to do this and we shall adopt one of the most traditional, based on the pairing functions π, π_1, π_2.

The empty sequence is coded by the number 0 and a sequence n_0, n_1, $\ldots$, n_{k-1} is coded by the "sequence number"

$$\langle n_0, n_1, \ldots, n_{k-1} \rangle = \pi'(\ldots \pi'(\pi'(0, n_0), n_1), \ldots, n_{k-1})$$

with $\pi'(a, b) := \pi(a, b) + 1$, thus recursively,

$$\begin{aligned} \langle\rangle &:= 0, \\ \langle n_0, n_1, \ldots, n_k \rangle &:= \pi'(\langle n_0, n_1, \ldots, n_{k-1} \rangle, n_k). \end{aligned}$$

Because of the surjectivity of π, every number a can be decoded uniquely as a sequence number $a = \langle n_0, n_1, \ldots, n_{k-1} \rangle$. If a is greater than zero, $\mathrm{hd}(a) := \pi_2(a \dot{-} 1)$ is the "head" (i.e., rightmost element) and $\mathrm{tl}(a) := \pi_1(a \dot{-} 1)$ is the "tail" of the list. The kth iterate of tl is denoted $\mathrm{tl}^{(k)}$ and since $\mathrm{tl}(a)$ is less than or equal to a, $\mathrm{tl}^{(k)}(a)$ is elementarily definable (by limited recursion). Thus we can define elementarily the "length" and "decoding" functions:

$$\mathrm{lh}(a) := \mu_{k \leq a}(\mathrm{tl}^{(k)}(a) = 0),$$

$$(a)_i \;:= \mathrm{hd}(\mathrm{tl}^{(\mathrm{lh}(a) \dot{-} (i+1))}(a)).$$

Then if $a = \langle n_0, n_1, \ldots, n_{k-1} \rangle$ it is easy to check that

$$\mathrm{lh}(a) = k \text{ and } (a)_i = n_i \text{ for each } i < k.$$

Furthermore $(a)_i = 0$ when $i \geq \mathrm{lh}(a)$. We shall write $(a)_{i,j}$ for $((a)_i)_j$ and $(a)_{i,j,k}$ for $(((a)_i)_j)_k$. This elementary coding machinery will be used at various crucial points in the following.

Note that our previous remarks show that the functions $\mathrm{lh}(\cdot)$ and $(a)_i$ are subelementary, and so is $\langle n_0, n_1, \ldots, n_{k-1} \rangle$ for each fixed k.

LEMMA (Estimate for sequence numbers).

$$(n+1)k \leq \langle \underbrace{n, \ldots, n}_{k} \rangle < (n+1)^{2^k}.$$

PROOF. We prove a slightly strengthened form of the second estimate:

$$\langle \underbrace{n, \ldots, n}_{k} \rangle + n + 1 \leq (n+1)^{2^k},$$

by induction on k. For $k = 0$ the claim is clear. In the step $k \mapsto k+1$ we have

$$\begin{aligned} \langle \underbrace{n, \ldots, n}_{k+1} \rangle + n + 1 &= \pi(\langle \underbrace{n, \ldots, n}_{k} \rangle, n) + n + 2 \\ &\leq (\langle \underbrace{n, \ldots, n}_{k} \rangle + n + 1)^2 \quad \text{by the lemma in 3.2.3} \\ &\leq (n+1)^{2^{k+1}} \quad \text{by induction hypothesis.} \end{aligned}$$

For the first estimate the base case $k = 0$ is clear, and in the step we have

$$\begin{aligned} \langle \underbrace{n, \ldots, n}_{k+1} \rangle &= \pi(\langle \underbrace{n, \ldots, n}_{k} \rangle, n) + 1 \\ &\geq \langle \underbrace{n, \ldots, n}_{k} \rangle + n + 1 \\ &\geq (n+1)(k+1) \quad \text{by induction hypothesis.} \end{aligned}$$

□

Concatenation of sequence numbers $b * a$ is defined thus:

$$\begin{aligned} b * \langle\rangle &:= b, \\ b * \langle n_0, n_1, \ldots, n_k \rangle &:= \pi(b * \langle n_0, n_1, \ldots, n_{k-1} \rangle, n_k) + 1. \end{aligned}$$

To check that this operation is also elementary, define $h(b, a, i)$ by recursion on i as follows.

$$h(b, a, 0) \quad = b,$$

$$h(b, a, i+1) = \pi(h(b, a, i), (a)_i) + 1$$

and note that since

$$h(b, a, i) = \langle (b)_0, \ldots, (b)_{\mathrm{lh}(b) \dot{-} 1}, (a)_0, \ldots, (a)_{i \dot{-} 1} \rangle \quad \text{for } i \leq \mathrm{lh}(a)$$

it follows from the estimate above that $h(a, b, i) \leq (b + a)^{2^{\mathrm{lh}(b)+i}}$. Thus h is definable by limited recursion from elementary functions and hence is itself elementary. Finally

$$b * a = h(b, a, \mathrm{lh}(a)).$$

LEMMA. *The class $\mathcal{E}$ is closed under limited course-of-values recursion. Thus if h, k are given functions in $\mathcal{E}$ and f is defined from them according to the schema*

$$f(\vec{m}, n) = h(n, \langle f(\vec{m}, 0), \ldots, f(\vec{m}, n-1) \rangle, \vec{m})$$
$$f(\vec{m}, n) \leq k(\vec{m}, n)$$

then f is in $\mathcal{E}$ also.

PROOF. $\bar{f}(\vec{m}, n) := \langle f(\vec{m}, 0), \ldots, f(\vec{m}, n-1) \rangle$ is definable by

$$\begin{aligned} \bar{f}(\vec{m}, 0) &= 0, \\ \bar{f}(\vec{m}, n+1) &= \bar{f}(\vec{m}, n) * \langle h(n, \bar{f}(\vec{m}, n), \vec{m}) \rangle \\ \bar{f}(\vec{m}, n) &\leq (\sum_{i \leq n} k(\vec{m}, i) + 1)^{2^n}, \end{aligned}$$

using $\langle \underbrace{n, \ldots, n}_{k} \rangle < (n+1)^{2^k}$. But $f(\vec{m}, n) = (\bar{f}(\vec{m}, n+1))_n$. □

The next lemma gives closure of $\mathcal{E}$ under limited course-of-values recursion but with parameter substitution allowed. Here we are working at the extremity of elementary definability, but this generalized schema will be crucially important for the elementary arithmetization of syntax which is developed prior to Gödel's theorems in the next chapter (particularly in regard to the substitution function). Unfortunately this last closure property of $\mathcal{E}$ is rather complicated to state, because it requires notational details to do with iteration of parameter substitutions.

LEMMA. *The class $\mathcal{E}$ is closed under limited course-of-values recursion with parameter substitution. Suppose g, h, k, p_i and a_i (for $i \leq l$) are all in $\mathcal{E}$ and let f be defined from them as follows.*

$$f(m, n) = \begin{cases} g(m) & \text{if } n = 0 \\ h(n, f(p_0(m, n), a_0(n)), \ldots, f(p_l(m, n), a_l(n)), m) & \text{otherwise} \end{cases}$$
$$f(m, n) \leq k(m, n)$$

where $a_i(n) < n$ when $n > 0$. Then f is also in $\mathcal{E}$ provided that the iterated parameter function $p(\sigma, m, n)$ defined below is elementarily bounded.

For any sequence $\sigma := \langle i_0, i_1, \ldots, i_{r-1}\rangle$ of numbers $\leq l$ define $n(\sigma)$ by: $n(\langle\rangle) := n$, $n(\sigma * \langle i\rangle) := a_i(n(\sigma))$ if $n(\sigma) \neq 0$ and $:= 0$ otherwise. Then $p(\sigma, m, n)$ is given by the course-of-values recursion:

$$p(\langle\rangle, m, n) = m$$

$$p(\sigma * \langle i\rangle, m, n) = \begin{cases} p_i(p(\sigma, m, n), n(\sigma)) & \text{if } n(\sigma) \neq 0 \\ p(\sigma, m, n) & \text{if } n(\sigma) = 0. \end{cases}$$

PROOF. First note that since $p(\sigma, m, n)$ is defined by a course-of-values recursion and, by supposition, is elementarily bounded, it is itself in $\mathcal{E}$ by the last lemma. Similarly, $n(\sigma)$ is elementary.

We code the computation of $f(m, n)$ as a finitely branching tree of height $\leq n + 1$. Nodes are sequence numbers $\sigma = \langle i_0, i_1, \ldots, i_{r-1}\rangle$ with $i_j \leq l$ and each such node is bounded in value by $(l+1)^{2^{n+1}}$. At each node σ is attached the value of f at the current parameter substitution $p(\sigma, m, n)$ and the current stage $n(\sigma)$. Let $Q(m, n, z)$ be the elementary relation expressing the fact that z correctly encodes the computation tree for $f(m, n)$ with $(z)_\sigma$ being the correct value at current node σ. Thus $Q(m, n, z)$ is the following condition, for all nodes $\sigma \leq (l+1)^{2^{n+1}}$: If $n(\sigma) \neq 0$, $(z)_\sigma = h(n(\sigma), (z)_{\sigma*\langle 0\rangle}, \ldots, (z)_{\sigma*\langle l\rangle}, p(\sigma, m, n))$ and if $n(\sigma) = 0$ then $(z)_\sigma = g(p(\sigma, m, n))$. Clearly Q is an elementary relation, and if z is the least such that $Q(m, n, z)$ holds then $f(m, n) = (z)_{\langle\rangle}$. Therefore f will be elementary if z can be bounded by an elementary function. This is now easy because $z = \langle (z)_{\langle\rangle}, (z)_1, \ldots (z)_{(l+1)^{2^{n+1}}}\rangle$ where each $(z)_\sigma = f(p(\sigma, m, n), n(\sigma)) \leq k(p(\sigma, m, n), n(\sigma))$. Therefore

$$z \leq (\max\{\, k(p(\sigma, m, n), n(\sigma)) \mid \sigma \leq (l+1)^{2^{n+1}} \,\} + 1)^{2^{(l+1)^{2^{n+1}}}}$$

and this is elementary. □

3.3. The Normal Form Theorem

3.3.1. Program numbers. The three types of register machine instructions I can be coded by "instruction numbers" $\sharp I$ thus, where v_0, v_1, v_2, ... is a list of all variables used to denote registers:

If I is "$v_j := 0$" then $\sharp I = \langle 0, j\rangle$.
If I is "$v_j := v_j + 1$" then $\sharp I = \langle 1, j\rangle$.
If I is "**if** $v_j = v_l$ **then** I_m **else** I_n" then $\sharp I = \langle 2, j, l, m, n\rangle$.

Clearly, using the sequence coding and decoding apparatus above, we can check elementarily whether or not a given number is an instruction number.

Any register machine program $P = I_0, I_1, \ldots, I_{k-1}$ can then be coded by a "program number" or "index" $\sharp P$ thus:

$$\sharp P = \langle \sharp I_0, \sharp I_1, \ldots, \sharp I_{k-1} \rangle$$

and again (although it is tedious) we can elementarily check whether or not a given number is indeed of the form $\sharp P$ for some program P. Tradition has it that e is normally reserved as a variable over putative program numbers.

Standard program constructs such as those in 3.1 have associated "index-constructors", i.e., functions which, given indices of the subprograms, produce an index for the constructed program. The point is that for standard program constructs the associated index-constructor functions are elementary. For example there is an elementary index-constructor comp such that, given programs P_0, P_1 with indices e_0, e_1, $\mathrm{comp}(e_0, e_1)$ is an index of the program P_0 **;** P_1. A moment's thought should convince the reader that the appropriate definition of comp is as follows:

$$\mathrm{comp}(e_0, e_1) = e_0 * \langle r(e_0, e_1, 0), r(e_0, e_1, 1), \ldots, r(e_0, e_1, \mathrm{lh}(e_1) \dot{-} 1) \rangle$$

where $r(e_0, e_1, i) =$

$$\begin{cases} \langle 2, (e_1)_{i,1}, (e_1)_{i,2}, (e_1)_{i,3} + \mathrm{lh}(e_0), (e_1)_{i,4} + \mathrm{lh}(e_0) \rangle & \text{if } (e_1)_{i,0} = 2 \\ (e_1)_i & \text{otherwise} \end{cases}$$

re-addresses the jump instructions in P_1. Clearly r and hence comp are elementary functions.

Definition. Henceforth, $\varphi_e^{(r)}$ denotes the partial function computed by the register machine program with program number e, operating on the input registers $v_1, \ldots, v_r$ and with output register v_0. There is no loss of generality here, since the variables in any program can always be renamed so that $v_1, \ldots, v_r$ become the input registers and v_0 the output. If e is not a program number, or it is but does not operate on the right variables, then we adopt the convention that $\varphi_e^{(r)}(n_1, \ldots, n_r)$ is undefined for all inputs $n_1, \ldots, n_r$.

3.3.2. Normal form.

Theorem (Kleene's Normal Form). *For each arity r there is an elementary function U and an elementary relation T such that, for all e and all inputs $n_1, \ldots, n_r$,*

(a) $\varphi_e^{(r)}(n_1, \ldots, n_r)$ *is defined if and only if* $\exists_s T(e, n_1, \ldots, n_r, s)$,

(b) $\varphi_e^{(r)}(n_1, \ldots, n_r) = U(e, n_1, \ldots, n_r, \mu_s T(e, n_1, \ldots, n_r, s))$.

Proof. A computation of a register machine program $P(v_1, \ldots, v_r; v_0)$ on numerical inputs $\vec{n} = n_1, \ldots, n_r$ proceeds deterministically, step by step,

each step corresponding to the execution of one instruction. Let e be its program number, and let $v_0, \ldots, v_l$ be all the registers used by P, including the "working registers" so $r \leq l$.

The "state" of the computation at step s is defined to be the sequence number

$$\text{state}(e, \vec{n}, s) = \langle e, i, m_0, m_1, \ldots, m_l \rangle$$

where $m_0, m_1, \ldots, m_l$ are the values stored in the registers $v_0, v_1, \ldots, v_l$ after step s is completed, and the next instruction to be performed is the ith one, thus $(e)_i$ is its instruction number.

The "state transition function" $\text{tr}\colon \mathbb{N} \to \mathbb{N}$ computes the "next state". So suppose that $x = \langle e, i, m_0, m_1, \ldots, m_l \rangle$ is any putative state. Then in what follows, $e = (x)_0$, $i = (x)_1$, and $m_j = (x)_{j+2}$ for each $j \leq l$. The definition of $\text{tr}(x)$ is therefore as follows:

$$\text{tr}(x) = \langle e, i', m_0', m_1', \ldots, m_l' \rangle$$

where

(i) If $(e)_i = \langle 0, j \rangle$ where $j \leq l$ then $i' = i + 1$, $m_j' = 0$, and all other registers remain unchanged, i.e., $m_k' = m_k$ for $k \neq j$.

(ii) If $(e)_i = \langle 1, j \rangle$ where $j \leq l$ then $i' = i + 1$, $m_j' = m_j + 1$, and all other registers remain unchanged.

(iii) If $(e)_i = \langle 2, j_0, j_1, i_0, i_1 \rangle$ where $j_0, j_1 \leq l$ and $i_0, i_1 \leq \text{lh}(e)$ then $i' = i_0$ or $i' = i_1$ according as $m_{j_0} = m_{j_1}$ or not, and all registers remain unchanged, i.e., $m_j' = m_j$ for all $j \leq l$.

(iv) Otherwise, if e is not a program number, or if it refers to a register v_k with $l < k$, or if $\text{lh}(e) \leq i$, then $\text{tr}(x)$ simply repeats the same state x so $i' = i$, and $m_j' = m_j$ for every $j \leq l$.

Clearly tr is an *elementary* function, since it is defined by elementarily decidable cases, with (a great deal of) elementary decoding and re-coding involved in each case.

Consequently, the "state function" $\text{state}(e, \vec{n}, s)$ is also *elementary* because it can be defined by iterating the transition function by limited recursion on s as follows:

$$\begin{aligned} \text{state}(e, \vec{n}, 0) &= \langle e, 0, n_1, \ldots, n_r, 0, \ldots, 0 \rangle \\ \text{state}(e, \vec{n}, s+1) &= \text{tr}(\text{state}(e, \vec{n}, s)) \\ \text{state}(e, \vec{n}, s) &\leq h(e, \vec{n}, s) \end{aligned}$$

where for the bounding function h we can take

$$h(e, \vec{n}, s) = \langle e, e \rangle * \langle \max(\vec{n}\,) + s, \ldots, \max(\vec{n}\,) + s \rangle.$$

This is because the maximum value of any register at step s cannot be greater than $\max(\vec{n}\,) + s$. Now this expression clearly is elementary, since

$\langle m, \ldots, m\rangle$ with i occurrences of m is definable by a limited recursion with bound $(m+i)^{2^i}$, as is easily seen by induction on i.

Now recall that if program P has program number e then computation terminates when instruction $I_{\mathrm{lh}(e)}$ is encountered. Thus we can define the "termination relation" $T(e, \vec{n}, s)$ meaning "computation terminates at step s", by

$$T(e, \vec{n}, s) := ((\mathrm{state}(e, \vec{n}, s))_1 = \mathrm{lh}(e)).$$

Clearly T is elementary and

$$\varphi_e^{(r)}(\vec{n}\,) \text{ is defined} \leftrightarrow \exists_s T(e, \vec{n}, s).$$

The output on termination is the value of register v_0, so if we define the "output function" $U(e, \vec{n}, s)$ by

$$U(e, \vec{n}, s) := (\mathrm{state}(e, \vec{n}, s))_2$$

then U is also elementary and

$$\varphi_e^{(r)}(\vec{n}\,) = U(e, \vec{n}, \mu_s T(e, \vec{n}, s)). \qquad \square$$

3.3.3. Σ_1^0-definable relations and μ-recursive functions. A relation R of arity r is said to be Σ_1^0-*definable* if there is an elementary relation E, say of arity $r+l$, such that for all $\vec{n} = n_1, \ldots, n_r$,

$$R(\vec{n}\,) \leftrightarrow \exists_{k_1} \ldots \exists_{k_l} E(\vec{n}, k_1, \ldots, k_l).$$

A partial function φ is said to be Σ_1^0-*definable* if its graph

$$\{\, (\vec{n}, m) \mid \varphi(\vec{n}\,) \text{ is defined and } = m \,\}$$

is Σ_1^0-definable.

To say that a non-empty relation R is Σ_1^0-definable is equivalent to saying that the set of all sequences $\langle \vec{n}\rangle$ satisfying R can be enumerated (possibly with repetitions) by some elementary function $f\colon \mathbb{N} \to \mathbb{N}$. Such relations are called *elementarily enumerable*. For choose any fixed sequence $\langle a_1, \ldots, a_r\rangle$ satisfying R and define

$$f(m) = \begin{cases} \langle (m)_1, \ldots, (m)_r\rangle & \text{if } E((m)_1, \ldots, (m)_{r+l}) \\ \langle a_1, \ldots, a_r\rangle & \text{otherwise.} \end{cases}$$

Conversely if R is elementarily enumerated by f then

$$R(\vec{n}\,) \leftrightarrow \exists_m (f(m) = \langle \vec{n}\rangle)$$

is a Σ_1^0-definition of R.

The *μ-recursive functions* are those (partial) functions which can be defined from the initial functions: constant 0, successor S, projections (onto the ith coordinate), addition $+$, modified subtraction $\dot{-}$ and multiplication

$\cdot$, by applications of composition and unbounded minimization. Note that it is through unbounded minimization that partial functions may arise.

LEMMA. *Every elementary function is μ-recursive.*

PROOF. By simply removing the bounds on μ in the lemmas in 3.2.3 one obtains μ-recursive definitions of the pairing functions π, π_1, π_2 and of Gödel's β-function. Then by removing all mention of bounds from the theorem in 3.2.4 one sees that the μ-recursive functions are closed under (unlimited) primitive recursive definitions: $f(\vec{m}, 0) = g(\vec{m})$, $f(\vec{m}, n+1) = h(n, f(\vec{m}, n))$. Thus one can μ-recursively define bounded sums and bounded products, and hence all elementary functions. □

3.3.4. Computable functions.

DEFINITION. The *while-programs* are those programs which can be built up from assignment statements $x := 0$, $x := y$, $x := y + 1$, $x := y \dot{-} 1$, by Conditionals, Composition, For-Loops and While-Loops as in 3.1 (on program constructs).

THEOREM. *The following are equivalent:*

(a) φ *is register machine computable,*
(b) φ *is Σ_1^0-definable,*
(c) φ *is μ-recursive,*
(d) φ *is computable by a while program.*

PROOF. The Normal Form Theorem shows immediately that every register machine computable function $\varphi_e^{(r)}$ is Σ_1^0-definable since

$$\varphi_e^{(r)}(\vec{n}\,) = m \leftrightarrow \exists_s \big(T(e, \vec{n}, s) \wedge U(e, \vec{n}, s) = m \big)$$

and the relation $T(e, \vec{n}, s) \wedge U(e, \vec{n}, s) = m$ is clearly elementary. If φ is Σ_1^0-definable, say

$$\varphi(\vec{n}\,) = m \leftrightarrow \exists_{k_1} \ldots \exists_{k_l} E(\vec{n}, m, k_1, \ldots, k_l)$$

then φ can be defined μ-recursively by

$$\varphi(\vec{n}\,) = (\mu_m E(\vec{n}, (m)_0, (m)_1, \ldots, (m)_l)\,)_0 \ ,$$

using the fact (above) that elementary functions are μ-recursive. The examples of computable functionals in 3.1 show how the definition of any μ-recursive function translates automatically into a while program. Finally, 3.1 shows how to implement any while program on a register machine. □

Henceforth *computable* means "register machine computable" or any of its equivalents.

COROLLARY. *The function $\varphi_e^{(r)}(n_1, \ldots, n_r)$ is a computable partial function of the $r+1$ variables $e, n_1, \ldots, n_r$.*

PROOF. Immediate from the Normal Form. □

LEMMA. *A relation R is computable if and only if both R and its complement $\mathbb{N}^n \setminus R$ are Σ_1^0-definable.*

PROOF. We can assume that both R and $\mathbb{N}^n \setminus R$ are not empty, and (for simplicity) also $n = 1$.

"$\rightarrow$". By the theorem above every computable relation is Σ_1^0-definable, and with R clearly its complement is computable.

"*gets*". Let $f, g \in \mathcal{E}$ enumerate R and $\mathbb{N} \setminus R$, respectively. Then

$$h(n) := \mu_i(f(i) = n \vee g(i) = n)$$

is a total μ-recursive function, and $R(n) \leftrightarrow f(h(n)) = n$. □

3.3.5. Undecidability of the halting problem. The above corollary says that there is a single "universal" program which, given numbers e and $\vec{n}$, computes $\varphi_e^{(r)}(\vec{n}\,)$ if it is defined. However we cannot decide in advance whether or not it will be defined. There is no program which, given e and $\vec{n}$, computes the total function

$$h(e, \vec{n}\,) = \begin{cases} 1 & \text{if } \varphi_e^{(r)}(\vec{n}\,) \text{ is defined,} \\ 0 & \text{if } \varphi_e^{(r)}(\vec{n}\,) \text{ is undefined.} \end{cases}$$

For suppose there were such a program. Then the function

$$\psi(\vec{n}\,) = \mu_m(h(n_1, \vec{n}\,) = 0)$$

would be computable, say with fixed program number e_0, and therefore

$$\varphi_{e_0}^{(r)}(\vec{n}\,) = \begin{cases} 0 & \text{if } h(n_1, \vec{n}\,) = 0 \\ \text{undefined} & \text{if } h(n_1, \vec{n}\,) = 1. \end{cases}$$

But then fixing $n_1 = e_0$ gives:

$$\varphi_{e_0}^{(r)}(\vec{n}\,) \text{ defined} \leftrightarrow h(e_0, \vec{n}\,) = 0 \leftrightarrow \varphi_{e_0}^{(r)}(\vec{n}\,) \text{ undefined},$$

a contradiction. Hence the relation $R(e, \vec{n}\,)$ which holds if and only if $\varphi_e^{(r)}(\vec{n}\,)$ is defined, is not recursive. It is however Σ_1^0-definable.

CHAPTER 4

Gödel's Theorems

4.1. Gödel Numbers

We will assign numbers – so-called Gödel numbers, GN for short – to the syntactical constructs developed in chapter 1: terms, formulas and derivations. Using the elementary sequence-coding and decoding machinery developed earlier we will be able to construct the code number of a composed object from its parts, and conversely to disassemble the code number of a composed object into the code numbers of its parts.

4.1.1. Gödel numbers of terms, formulas and derivations. Let $\mathcal{L}$ be a countable first order language. Assume that we have injectively assigned to every n-ary relation symbol R a *symbol number* $\mathrm{SN}(R)$ of the form $\langle 1, n, i\rangle$ and to every n-ary function symbol f a symbol number $\mathrm{SN}(f)$ of the form $\langle 2, n, j\rangle$. Call $\mathcal{L}$ *elementarily presented* if the set $\mathrm{Symb}_{\mathcal{L}}$ of all these symbol numbers is elementary. In what follows we shall always assume that the languages $\mathcal{L}$ considered are elementarily presented. In particular this applies to every language with finitely many relation and function symbols.

Let $\mathrm{SN}(\mathrm{Var}) := \langle 0\rangle$. For every $\mathcal{L}$-term r we define recursively its Gödel number $\ulcorner r \urcorner$ by

$$\begin{aligned} \ulcorner x_i \urcorner &:= \langle \mathrm{SN}(\mathrm{Var}), i\rangle, \\ \ulcorner f r_1 \dots r_n \urcorner &:= \langle \mathrm{SN}(f), \ulcorner r_1 \urcorner, \dots, \ulcorner r_n \urcorner\rangle. \end{aligned}$$

Assign numbers to the logical symbols by $\mathrm{SN}(\to) := \langle 3, 0\rangle$ und $\mathrm{SN}(\forall) := \langle 3, 1\rangle$. For simplicity we leave out the logical connectives $\wedge$, $\vee$ and $\exists$ here; they could be treated similarly. We define for every $\mathcal{L}$-formula A its Gödel number $\ulcorner A \urcorner$ by

$$\begin{aligned} \ulcorner R r_1 \dots r_n \urcorner &:= \langle \mathrm{SN}(R), \ulcorner r_1 \urcorner, \dots, \ulcorner r_n \urcorner\rangle, \\ \ulcorner A \to B \urcorner &:= \langle \mathrm{SN}(\to), \ulcorner A \urcorner, \ulcorner B \urcorner\rangle, \\ \ulcorner \forall_{x_i} A \urcorner &:= \langle \mathrm{SN}(\forall), i, \ulcorner A \urcorner\rangle. \end{aligned}$$

We define symbol numbers for the names of the natural deduction rules: $\mathrm{SN}(\mathrm{AssVar}) := \langle 4, 0\rangle$, $\mathrm{SN}(\to^+) := \langle 4, 1\rangle$, $\mathrm{SN}(\to^-) := \langle 4, 2\rangle$, $\mathrm{SN}(\forall^+) :=$

$\langle 4,3\rangle$, $\mathrm{SN}(\forall^-) := \langle 4,4\rangle$. For a derivation M we define its Gödel number $\ulcorner M\urcorner$ by

$$\begin{aligned}
\ulcorner u_i^A\urcorner &:= \langle \mathrm{SN}(\mathrm{AssVar}), i, \ulcorner A\urcorner\rangle,\\
\ulcorner \lambda_{u_i^A} M\urcorner &:= \langle \mathrm{SN}(\to^+), i, \ulcorner A\urcorner, \ulcorner M\urcorner\rangle,\\
\ulcorner MN\urcorner &:= \langle \mathrm{SN}(\to^-), \ulcorner M\urcorner, \ulcorner N\urcorner\rangle,\\
\ulcorner \lambda_{x_i} M\urcorner &:= \langle \mathrm{SN}(\forall^+), i, \ulcorner M\urcorner\rangle,\\
\ulcorner Mr\urcorner &:= \langle \mathrm{SN}(\forall^-), \ulcorner M\urcorner, \ulcorner r\urcorner\rangle.
\end{aligned}$$

It will be helpful in the sequel to have some general estimates on Gödel numbers, which we provide here. For a term r or formula A we define its *sum of maximal sequence lengths* $\|r\|$ or $\|A\|$ by

$$\begin{aligned}
&\|x_i\| := 2,\\
&\|fr_0\dots r_{k-1}\| := k+1+\max(\|r_i\|),
\end{aligned}
\qquad
\begin{aligned}
&\|Rr_0\dots r_{k-1}\| := k+1+\max(\|r_i\|),\\
&\|A\to B\| := 3+\max(\|A\|,\|B\|),\\
&\|\forall_x A\| := 3+\|A\|
\end{aligned}$$

and its *symbol bound* $\mathrm{Symb}(r)$ or $\mathrm{Symb}(A)$ by

$$\begin{aligned}
&\mathrm{Symb}(x_i) := \max(\mathrm{SN}(\mathrm{Var}), i)+1,\\
&\mathrm{Symb}(fr_0\dots r_{k-1}) := \max(\mathrm{SN}(f), \max(\mathrm{Symb}(r_i))),\\
&\mathrm{Symb}(Rr_0\dots r_{k-1}) := \max(\mathrm{SN}(R), \max(\mathrm{Symb}(r_i))),\\
&\mathrm{Symb}(A\to B) := \max(\mathrm{Symb}(A), \mathrm{Symb}(B)),\\
&\mathrm{Symb}(\forall_x A) := \mathrm{Symb}(A).
\end{aligned}$$

LEMMA. *$\|r\| \le \ulcorner r\urcorner < \mathrm{Symb}(r)^{2^{\|r\|}}$ and $\|A\| \le \ulcorner A\urcorner < \mathrm{Symb}(A)^{2^{\|A\|}}$.*

PROOF. We prove $\|r\| \le \ulcorner r\urcorner$ by induction on r. The case of a variable x_i is easy:

$$\|x_i\| = 2 \le \langle \mathrm{SN}(\mathrm{Var}), i\rangle = \ulcorner x_i\urcorner,$$

and for a term $fr_0\dots r_{k-1}$ we have by the sequence coding,

$$\begin{aligned}
\ulcorner fr_0\dots r_{k-1}\urcorner &= \langle \mathrm{SN}(f), \ulcorner r_0\urcorner, \dots, \ulcorner r_{k-1}\urcorner\rangle\\
&\ge k+1+\max(\ulcorner r_i\urcorner)\\
&\ge k+1+\max(\|r_i\|) \quad \text{by induction hypothesis}\\
&= \|fr_0\dots r_{k-1}\|.
\end{aligned}$$

The proof of $\|A\| \le \ulcorner A\urcorner$ is similar. For $\ulcorner r\urcorner < \mathrm{Symb}(r)^{2^{\|r\|}}$ we again use induction on r. For a variable x_i we obtain by the estimate in 3.2.5

$$\ulcorner x_i\urcorner = \langle \mathrm{SN}(\mathrm{Var}), i\rangle < \mathrm{Symb}(x_i)^{2^2} = \mathrm{Symb}(x_i)^{2^{\|x_i\|}}$$

and for a term $r := fr_0 \dots r_{k-1}$ built with a function symbol f we have

$$\begin{aligned}
&\ulcorner fr_0 \dots r_{k-1} \urcorner \\
&= \langle \mathrm{SN}(f), \ulcorner r_0 \urcorner, \dots, \ulcorner r_{k-1} \urcorner \rangle \\
&\leq \underbrace{\langle n \dot{-} 1, n \dot{-} 1, \dots, n \dot{-} 1 \rangle}_{k+1} \quad \text{with } n := \mathrm{Symb}(r)^{2^{\max \|r_i\|}}, \text{ by ind. hyp.} \\
&< n^{2^{k+1}} \quad \text{by the estimate in 3.2.5} \\
&= \mathrm{Symb}(r)^{2^{k+1+\max \|r_i\|}} = \mathrm{Symb}(r)^{2^{\|r\|}}.
\end{aligned}$$

The proof of $\ulcorner A \urcorner < \mathrm{Symb}(A)^{2^{\|A\|}}$ is again similar, but we spell out the quantifier case $A = \forall_{x_i} B$:

$$\ulcorner A \urcorner = \langle \mathrm{SN}(\forall), i, \ulcorner B \urcorner \rangle \leq \max(\mathrm{SN}(\forall), i, \mathrm{Symb}(B)^{2^{\|B\|}})^{2^3} \leq \mathrm{Symb}(A)^{2^{\|A\|}}. \ \square$$

4.1.2. Elementary functions on Gödel numbers. We shall define an elementary predicate Deriv such that $\mathrm{Deriv}(d)$ if and only if d is the Gödel number of a derivation. To this end we need a number of auxiliary functions and relations, which will all be elementary and have the properties described. (The convention is that relations are capitalized and functions are lower case). First we need some basic notions:

$\mathrm{Ter}(t)$	t is GN of a term,
$\mathrm{For}(a)$	a is GN of a formula,
$\mathrm{FV}(i, y)$	the variable x_i is free in the term or formula with GN y,
$\mathrm{fmla}(d)$	GN of the formula derived by the derivation with GN d.

By the *context* of a derivation M we mean the set $\{u_{i_0}^{A_0}, \dots, u_{i_{n-1}}^{A_{n-1}}\}$ of its free assumption variables, where $i_0 < \dots < i_{n-1}$. Its Gödel number is defined to be the least number c such that $\forall_{\nu<n}((c)_{i_\nu} = \ulcorner A_\nu \urcorner)$.

$\mathrm{ctx}(d)$	GN of the context of the derivation with GN d,
$\mathrm{Cons}(c_1, c_2)$	the contexts with GN c_1, c_2 are consistent.

Then Deriv can be defined by course-of-values recursion, using the next-to-last lemma in 3.2.5.

$$\begin{aligned}
\mathrm{Deriv}(d) := {} & ((d)_0 = \mathrm{SN}(\mathrm{AssVar}) \wedge \mathrm{lh}(d) = 3 \wedge \mathrm{For}((d)_2)) \vee \\
& ((d)_0 = \mathrm{SN}(\rightarrow^+) \wedge \mathrm{lh}(d) = 4 \wedge \mathrm{For}((d)_2) \wedge \mathrm{Deriv}((d)_3) \wedge \\
& \quad ((\mathrm{ctx}((d)_3))_{(d)_1} \neq 0 \rightarrow (\mathrm{ctx}((d)_3))_{(d)_1} = (d)_2)) \vee \\
& ((d)_0 = \mathrm{SN}(\rightarrow^-) \wedge \mathrm{lh}(d) = 3 \wedge \mathrm{Deriv}((d)_1) \wedge \mathrm{Deriv}((d)_2) \wedge \\
& \quad \mathrm{Cons}(\mathrm{ctx}((d)_1), \mathrm{ctx}((d)_2)) \wedge
\end{aligned}$$

$$\begin{aligned}
&(\mathrm{fmla}((d)_1))_0 = \mathrm{SN}(\to) \wedge (\mathrm{fmla}((d)_1))_1 = \mathrm{fmla}((d)_2)) \vee \\
&((d)_0 = \mathrm{SN}(\forall^+) \wedge \mathrm{lh}(d) = 3 \wedge \mathrm{Deriv}((d)_2) \wedge \forall_{i<\mathrm{lh}(\mathrm{ctx}((d)_2))}(\\
&\quad (\mathrm{ctx}((d)_2))_i \neq 0 \to \neg \mathrm{FV}((d)_1, (\mathrm{ctx}((d)_2))_i))) \vee \\
&((d)_0 = \mathrm{SN}(\forall^-) \wedge \mathrm{lh}(d) = 3 \wedge \mathrm{Deriv}((d)_1) \wedge \mathrm{Ter}((d)_2) \wedge \\
&\quad (\mathrm{fmla}((d)_1)_0 = \mathrm{SN}(\forall)).
\end{aligned}$$

Still further auxiliary functions are needed. A *substitution* is a map $x_{i_0} \mapsto r_0, \ldots, x_{i_{n-1}} \mapsto r_{n-1}$ with $i_0 < \cdots < i_{n-1}$ from variables to terms; its Gödel number is the least number s such that $\forall_{\nu<n}((s)_{i_\nu} = \ulcorner r_\nu \urcorner)$. Hence $(s)_{i_\nu} = 0$ indicates that s leaves x_{i_ν} unchanged.

$\mathrm{union}(c_1, c_2)$	GN of the union of the consistent contexts with GN c_1, c_2,
$\mathrm{remove}(c, i)$	GN of result of removing u_i from the context with GN c,
$\mathrm{sub}(x, s)$	GN of the result of applying the substitution with GN s to the term or formula with GN x,
$\mathrm{update}(s, i, t)$	GN of the result of updating the substitution with GN s by changing its entry at i to the term with GN t.

We now give definitions of all these; from the form of the definitions it will be clear that they have the required properties, and are elementary.

Update. This can be defined explicitly, using the bounded least number operator:

$$\begin{aligned}
&\mathrm{update}(s, i, t) := \\
&\quad \mu_{x<h(\max(s,t),\max(\mathrm{lh}(s),i))}((x)_i = t \wedge \forall_{k<\max(\mathrm{lh}(s),i)}(k \neq i \to (x)_k = (s)_k))
\end{aligned}$$

where $h(n, k) := (n + 1)^{2^k}$ is the elementary function defined earlier with the property $\langle n, \ldots, n\rangle \leq h(n, k)$.

Substitution. The substitution function defined next takes a formula or term with GN x and applies to it a substitution with GN s to produce a new formula with GN y. The substitution works by assigning specific terms to the free variables, but in order to avoid clashing it must also reassign new variables to the universally bound ones. This occurs in the final clause of the definition where, to be on the safe side, we (recursively) assign to a bound variable the new variable with index $x + i(s)$, where $i(s)$ is the maximum index of any variable occurring in a value term $(s)_j$ of s. We define substitution by a limited course-of-values recursion with parameter substitutions:

$$\mathrm{sub}(x, s) :=$$

$$\begin{cases} x & \text{if } (x)_0 = \mathrm{SN}(\mathrm{Var}) \wedge (s)_{(x)_1} = 0 \\ (s)_{(x)_1} & \text{if } (x)_0 = \mathrm{SN}(\mathrm{Var}) \wedge (s)_{(x)_1} \neq 0 \\ \mu_{y \leq k(x,s)}(\mathrm{lh}(x) = \mathrm{lh}(y) \wedge (x)_0 = (y)_0 \wedge \forall_{i<l}(\mathrm{sub}((x)_{i+1}, s) = (y)_{i+1})) & \\ \quad \text{if } (x)_{0,0} = 1 \vee (x)_{0,0} = 2 \vee (x)_0 = \mathrm{SN}(\rightarrow) & \\ \langle \mathrm{SN}(\forall), x + i(s), \mathrm{sub}((x)_2, \mathrm{update}(s, (x)_1, \langle \mathrm{SN}(\mathrm{Var}), x + i(s) \rangle)) \rangle & \\ \quad \text{if } (x)_0 = \mathrm{SN}(\forall) & \\ 0 \quad \text{otherwise} & \end{cases}$$

$$\mathrm{sub}(x, s) \leq k(x, s)$$

where it is assumed that the relation and function symbols in the given language $\mathcal{L}$ all have arity $\leq l$. The bound $k(x, s)$ and a bound for the iterated parameter updates remain to be provided, so that the last lemma in 3.2.5 can be applied. Then sub will be elementary.

First notice that as s is continually updated by the recursion, for the sake of (the formula or term with GN) x, the first update assigns to a bound variable in x a "new" variable with index $x+i(s)$. The next update will then assign to a bound variable in some subformula x' of x a new variable with index $x' + x + i(s)$ etcetera. The final update will therefore be a sequence of length $\leq x^2+i(s)$, whose entries are all $< \max(s, \langle \mathrm{SN}(\mathrm{Var}), x^2+i(s) \rangle)$. Thus a bound for all iterated updates starting from s and x is this last expression to the power of $2^{x^2+i(s)}$, which is elementary.

Using the lemma in 4.1.1 above one can see that if x is the GN of a term or a formula X and s is the GN of a substitution S, so that we may write $\mathrm{sub}(x, s) = {}^{\ulcorner}X[S]^{\urcorner}$, then $\mathrm{Symb}(X[S]) \leq \max(s, x, x^2 + i(s)) \leq x^2 + s$ and, clearly, $\|X[S]\| \leq x + s$. The lemma then gives an elementary bound $k(x, s) := (x^2 + s)^{2^{x+s}}$ for $\mathrm{sub}(x, s)$.

Remove, union, consistency, context. Removal of an assumption variable from a context is defined by

$$\mathrm{remove}(c, i) := \mu_{x \leq c}((x)_i = 0 \wedge \forall_{j<\mathrm{lh}(c)}(j \neq i \rightarrow (x)_j = (c)_j)).$$

The union of two consistent contexts can again be defined by the bounded μ-operator:

$$\mathrm{union}(c_1, c_2) := \mu_{c \leq c_1 * c_2} \forall_{i<\max(\mathrm{lh}(c_1), \mathrm{lh}(c_2))}((c)_i = \max((c_1)_i, (c_2)_i)).$$

Consistency of two contexts is defined by

$$\mathrm{Cons}(c_1, c_2) := \forall_{i<\max(\mathrm{lh}(c_1), \mathrm{lh}(c_2))}((c_1)_i \neq 0 \rightarrow (c_2)_i \neq 0 \rightarrow (c_1)_i = (c_2)_i).$$

The context of a derivation is defined by

$$\begin{aligned} \mathrm{ctx}(d) := \mu_{c \leq d}(((d)_0 = \mathrm{SN}(\mathrm{AssVar}) \wedge (c)_{(d)_1} = (d)_2) \vee \\ ((d)_0 = \mathrm{SN}(\rightarrow^+) \wedge c = \mathrm{remove}(\mathrm{ctx}((d)_3), (d)_1)) \vee \end{aligned}$$

$$\begin{aligned}&((d)_0 = \mathrm{SN}(\to^-) \wedge c = \mathrm{union}(\mathrm{ctx}((d)_1), \mathrm{ctx}((d)_2))) \vee\\ &((d)_0 = \mathrm{SN}(\forall^+) \wedge c = \mathrm{ctx}((d)_2)) \vee\\ &((d)_0 = \mathrm{SN}(\forall^-) \wedge c = \mathrm{ctx}((d)_1))).\end{aligned}$$

Formulas, free variables, terms. The end formula of a derivation is defined by

$$\begin{aligned}\mathrm{fmla}(d) := \mu_{a\le d^{2^d}}&(((d)_0 = \mathrm{SN}(\mathrm{AssVar}) \wedge a = (d)_2) \vee\\ &((d)_0 = \mathrm{SN}(\to^+) \wedge a = \langle \mathrm{SN}(\to), (d)_2, \mathrm{fmla}((d)_3)\rangle) \vee\\ &((d)_0 = \mathrm{SN}(\to^-) \wedge a = (\mathrm{fmla}((d)_1))_2) \vee\\ &((d)_0 = \mathrm{SN}(\forall^+) \wedge a = \langle \mathrm{SN}(\forall), (d)_1, \mathrm{fmla}((d)_2)\rangle) \vee\\ &((d)_0 = \mathrm{SN}(\forall^-) \wedge\\ &\qquad \mathrm{sub}((\mathrm{fmla}((d)_1))_2, \mu_{s\le d}((s)_{(\mathrm{fmla}((d)_1))_1} = (d)_2)) = a)).\end{aligned}$$

Notice that this is the only place in our definitions of auxiliary functions and relations where the substitution function is needed.

Freeness of a variable x_i in a term or formula is defined by

$$\begin{aligned}\mathrm{FV}(i, y) := &((y)_0 = \mathrm{SN}(\mathrm{Var}) \wedge (y)_1 = i) \vee\\ &((y)_{0,0} = 1 \wedge \exists_{j<\mathrm{lh}(y)\dot{-}1}\mathrm{FV}(i, (y)_{j+1})) \vee\\ &((y)_{0,0} = 2 \wedge \exists_{j<\mathrm{lh}(y)\dot{-}1}\mathrm{FV}(i, (y)_{j+1})) \vee\\ &((y)_0 = \mathrm{SN}(\to) \wedge (\mathrm{FV}(i, (y)_1) \vee \mathrm{FV}(i, (y)_2))) \vee\\ &((y)_0 = \mathrm{SN}(\forall) \wedge i \neq (y)_1 \wedge \mathrm{FV}(i, (y)_2)).\end{aligned}$$

The sets of formulas and terms are defined by

$$\begin{aligned}&\mathrm{For}(a) :=\\ &((a)_{0,0} = 1 \wedge \mathrm{Symb}_{\mathcal{L}}((a)_0) \wedge \mathrm{lh}(a) = (a)_{0,1} + 1 \wedge \forall_{j<(a)_{0,1}}\mathrm{Ter}((a)_{j+1})) \vee\\ &((a)_0 = \mathrm{SN}(\to) \wedge \mathrm{lh}(a) = 3 \wedge \mathrm{For}((a)_1) \wedge \mathrm{For}((a)_2)) \vee\\ &((a)_0 = \mathrm{SN}(\forall) \wedge \mathrm{lh}(a) = 3 \wedge \mathrm{For}((a)_2)),\end{aligned}$$

$$\begin{aligned}\mathrm{Ter}(t) := &((t)_0 = \mathrm{SN}(\mathrm{Var}) \wedge \mathrm{lh}(t) = 2) \vee\\ &((t)_{0,0} = 2 \wedge \mathrm{Symb}_{\mathcal{L}}((t)_0) \wedge \mathrm{lh}(t) = (t)_{0,1}{+}1 \wedge \forall_{j<(t)_{0,1}}\mathrm{Ter}((t)_{j+1})).\end{aligned}$$

Recall that for simplicity we have left out the logical connectives $\wedge$, $\vee$ and $\exists$. They could be added easily, including an extension of the notion of a derivation to also allow their axioms as listed in 1.1.7.

4.1.3. Axiomatized theories. Call a relation *recursive* if its (total) characteristic function is recursive. A set S of formulas is called *recursive* (*elementary*, *primitive recursive*, *recursively enumerable*), if $\ulcorner S \urcorner := \{ \ulcorner A \urcorner \mid A \in S \}$ is recursive (elementary, primitive recursive, recursively enumerable). Clearly the sets $\mathrm{Stab}_{\mathcal{L}}$ of stability axioms and $\mathrm{Eq}_{\mathcal{L}}$ of $\mathcal{L}$-equality axioms are elementary. Now let $\mathcal{L}$ be an elementarily presented language with $=$ in $\mathcal{L}$. A theory T with $L(T) \subseteq \mathcal{L}$ is *recursively (elementarily, primitive recursively) axiomatizable*, if there is a recursive (elementary, primitive recursive) set S of closed $\mathcal{L}$-formulas such that $T = \{ A \in \overline{\mathcal{L}} \mid S \cup \mathrm{Eq}_{\mathcal{L}} \vdash A \}$.

THEOREM. *For theories T with $L(T) \subseteq \mathcal{L}$ the following are equivalent.*

(a) *T is recursively axiomatizable.*
(b) *T is primitive recursively axiomatizable.*
(c) *T is elementarily axiomatizable.*
(d) *T is recursively enumerable.*

PROOF. (d) $\to$ (c). Let $\ulcorner T \urcorner$ be recursively enumerable. Then there is an elementary f such that $\ulcorner T \urcorner = \mathrm{ran}(f)$. Let $f(n) = \ulcorner A_n \urcorner$. We define an elementary function g with the property $g(n) = \ulcorner A_0 \wedge \cdots \wedge A_n \urcorner$ by

$$\begin{aligned} g(0) &:= f(0), \\ g(n+1) &:= g(n) \mathbin{\dot\wedge} f(n+1), \\ g(n) &\leq \max(\mathrm{SN}(\wedge), m)^{2^{4n+m}} \quad \text{where } m := \max_{i<n} f(i) \end{aligned}$$

with $a \mathbin{\dot\wedge} b := \langle \mathrm{SN}(\wedge), a, b \rangle$. For $S := \{ A_0 \wedge \cdots \wedge A_n \mid n \in \mathbb{N} \}$ we have $\ulcorner S \urcorner = \mathrm{ran}(g)$, and this set is elementary because of $a \in \mathrm{ran}(g) \leftrightarrow \exists_{n<a}(a = g(n))$. T is elementarily axiomatizable, since $T = \{ A \in \overline{\mathcal{L}} \mid S \cup \mathrm{Eq}_{\mathcal{L}} \vdash A \}$.

(c) $\to$ (b) and (b) $\to$ (a) are clear.

(a) $\to$ (d). Let T be axiomatized by S with $\ulcorner S \urcorner$ recursive. Then

$$\begin{aligned} a \in \ulcorner T \urcorner \leftrightarrow \exists_d(&\mathrm{Deriv}(d) \wedge \mathrm{fmla}(d) = a \wedge \forall_{i<a} \neg \mathrm{FV}(i,a) \wedge \\ &\forall_{i<\mathrm{lh}(\mathrm{ctx}(d))}((\mathrm{ctx}(d))_i \in \ulcorner \mathrm{Eq}_{\mathcal{L}} \urcorner \cup \ulcorner S \urcorner)). \end{aligned}$$

Hence $\ulcorner T \urcorner$ is recursively enumerable. □

Call a theory T in our elementarily presented language $\mathcal{L}$ *axiomatized* if it is given by a recursively enumerable axiom system Ax_T. By the theorem just proved we can even assume that Ax_T is elementary. For such axiomatized theories we define a binary relation Prf_T by

$$\mathrm{Prf}_T(d,a) := \mathrm{Deriv}(d) \wedge \mathrm{fmla}(d) = a \wedge \forall_{i<\mathrm{lh}(\mathrm{ctx}(d))}((\mathrm{ctx}(d))_i \in \ulcorner \mathrm{Eq}_{\mathcal{L}} \urcorner \cup \ulcorner \mathrm{Ax}_T \urcorner).$$

Clearly Prf_T is elementary and $\mathrm{Prf}_T(d,a)$ if and only if d is the GN of a derivation of the formula with GN a from a context composed of equality axioms and formulas from Ax_T. A theory T is *consistent* if $\bot \notin T$; otherwise

T is *inconsistent*. A theory T is *complete* if for every closed formula A we have $A \in T$ or $A \notin T$, and *incomplete* otherwise.

COROLLARY. *Let T be a consistent theory. If T is axiomatized and complete then T is recursive.*

PROOF. We define the characteristic function $c_{\ulcorner T \urcorner}$ of $\ulcorner T \urcorner$ as follows. $c_{\ulcorner T \urcorner}(a)$ is 0 if $\neg \mathrm{For}(a)$ or $\exists_{i<a} \mathrm{FV}(i,a)$. Otherwise it is defined by

$$c_{\ulcorner T \urcorner}(a) = (\mu_x((\mathrm{Prf}_T((x)_0, a) \wedge (x)_1 = 1) \vee (\mathrm{Prf}_T((x)_0, \dot{\neg} a) \wedge (x)_1 = 0)))_1$$

with $\dot{\neg} a := \langle \mathrm{SN}(\rightarrow), a, \mathrm{SN}(\bot) \rangle$. Completeness of T implies that $c_{\ulcorner T \urcorner}$ is total, and consistency that it indeed is the characteristic function of $\ulcorner T \urcorner$. □

4.1.4. Undefinability of the notion of truth. Let $\mathcal{M}$ be an $\mathcal{L}$-structure. A relation $R \subseteq |\mathcal{M}|^n$ is called *definable* in $\mathcal{M}$ if there is an $\mathcal{L}$-formula $A(x_1, \ldots, x_n)$ such that

$$R = \{\, (a_1, \ldots, a_n) \in |\mathcal{M}|^n \mid \mathcal{M} \models A(x_1, \ldots x_n)[x_1 := a_1, \ldots, x_n := a_n] \,\}.$$

We assume in this section that $|\mathcal{M}| = \mathbb{N}$, 0 is a constant in $\mathcal{L}$ and S is a unary function symbol in $\mathcal{L}$ with $0^{\mathcal{M}} = 0$ and $\mathrm{S}^{\mathcal{M}}(a) = a + 1$. Recall that for every $a \in \mathbb{N}$ the *numeral* $\underline{a} \in \mathrm{Ter}_{\mathcal{L}}$ is defined by $\underline{0} := 0$ and $\underline{n+1} := \mathrm{S}\underline{n}$. Observe that in this case the definability of $R \subseteq \mathbb{N}^n$ by $A(x_1, \ldots, x_n)$ is equivalent to

$$R = \{\, (a_1, \ldots, a_n) \in \mathbb{N}^n \mid \mathcal{M} \models A(\underline{a_1}, \ldots, \underline{a_n}) \,\}.$$

Furthermore let $\mathcal{L}$ be an elementarily presented language. We will always assume in this chapter that every elementary relation is definable in $\mathcal{M}$. A set S of formulas is called *definable* in $\mathcal{M}$ if $\ulcorner S \urcorner := \{\, \ulcorner A \urcorner \mid A \in S \,\}$ is.

We shall show that already from these assumptions it follows that the notion of truth for $\mathcal{M}$, more precisely the set $\mathrm{Th}(\mathcal{M})$ of all closed formulas valid in $\mathcal{M}$, is undefinable in $\mathcal{M}$. From this it will follow that the notion of truth is in fact undecidable, for otherwise the set $\mathrm{Th}(\mathcal{M})$ would be recursive (Church's Thesis), hence recursively enumerable, and hence definable, because we have assumed already that all elementary relations are definable in $\mathcal{M}$ and so their projections are definable also. For the proof we shall need the following Fixed Point Lemma, which will be generalized in 4.2.2.

LEMMA (Semantical Fixed Point Lemma). *If every elementary relation is definable in $\mathcal{M}$, then for every $\mathcal{L}$-formula $B(z)$ we can find a closed $\mathcal{L}$-formula A such that*

$$\mathcal{M} \models A \quad \textit{if and only if} \quad \mathcal{M} \models B(\underline{\ulcorner A \urcorner}).$$

PROOF. Let s be the elementary function satisfying for every formula $C = C(z)$ with $z := x_0$,

$$s(\ulcorner C \urcorner, k) = \mathrm{sub}(\ulcorner C \urcorner, \langle \ulcorner \underline{k} \urcorner \rangle) = \ulcorner C(\underline{k}) \urcorner$$

where sub is the substitution function already defined in 4.1.2. Hence in particular

$$s(\ulcorner C \urcorner, \ulcorner C \urcorner) = \ulcorner C(\underline{\ulcorner C \urcorner}) \urcorner.$$

By assumption the graph G_s of s is definable in $\mathcal{M}$, by $A_s(x_1, x_2, x_3)$ say. Let

$$C := \exists_x (B(x) \wedge A_s(z, z, x)), \quad A := C(\underline{\ulcorner C \urcorner}),$$

and therefore

$$A = \exists_x (B(x) \wedge A_s(\underline{\ulcorner C \urcorner}, \underline{\ulcorner C \urcorner}, x)).$$

Hence $\mathcal{M} \models A$ if and only if $\exists_{a \in \mathbb{N}}((\mathcal{M} \models B(\underline{a})) \wedge a = \ulcorner C(\underline{\ulcorner C \urcorner}) \urcorner)$, which is the same as $\mathcal{M} \models B(\underline{\ulcorner A \urcorner})$. □

THEOREM (Tarski's Undefinability Theorem). *Assume that every elementary relation is definable in $\mathcal{M}$. Then* $\mathrm{Th}(\mathcal{M})$ *is undefinable in $\mathcal{M}$, hence in particular not recursively enumerable.*

PROOF. Assume that $\ulcorner \mathrm{Th}(\mathcal{M}) \urcorner$ is definable by $B_W(z)$. Then for all closed formulas A

$$\mathcal{M} \models A \quad \text{if and only if} \quad \mathcal{M} \models B_W(\underline{\ulcorner A \urcorner}).$$

Now consider the formula $\neg B_W(z)$ and choose by the Fixed Point Lemma a closed $\mathcal{L}$-formula A such that

$$\mathcal{M} \models A \quad \text{if and only if} \quad \mathcal{M} \models \neg B_W(\underline{\ulcorner A \urcorner}).$$

This contradicts the equivalence above.

We already have noticed that all recursively enumerable relations are definable in $\mathcal{M}$. Hence it follows that $\ulcorner \mathrm{Th}(\mathcal{M}) \urcorner$ cannot be recursively enumerable. □

4.2. The Notion of Truth in Formal Theories

We now want to generalize the arguments of the previous section. There we have made essential use of the notion of truth in a structure $\mathcal{M}$, i.e., of the relation $\mathcal{M} \models A$. The set of all closed formulas A such that $\mathcal{M} \models A$ has been called the theory of $\mathcal{M}$, denoted $\mathrm{Th}(\mathcal{M})$.

Now instead of $\mathrm{Th}(\mathcal{M})$ we shall start more generally from an arbitrary theory T. We consider the question as to whether in T there is a *notion of truth* (in the form of a *truth formula* $B(z)$), such that $B(z)$ "means" that z is "true". A consequence is that we have to explain all the notions used without referring to semantical concepts at all.

(i) z ranges over closed formulas (or sentences) A, or more precisely over their Gödel numbers $\ulcorner A \urcorner$.
(ii) A "true" is to be replaced by $T \vdash A$.
(iii) C "equivalent" to D is to be replaced by $T \vdash C \leftrightarrow D$.

Hence the question now is whether there is a truth formula $B(z)$ such that $T \vdash A \leftrightarrow B(\underline{\ulcorner A \urcorner})$ for all sentences A. The result will be that this is impossible, under rather weak assumptions on the theory T. Technically, the issue will be to replace the notion of definability by the notion of "representability" within a formal theory. We begin with a discussion of this notion.

In this section we assume that $\mathcal{L}$ is an elementarily presented language with 0, S and $=$ in $\mathcal{L}$, and T an $\mathcal{L}$-theory containing the equality axioms $\mathrm{Eq}_{\mathcal{L}}$.

4.2.1. Representable relations and functions.

Definition. A relation $R \subseteq \mathbb{N}^n$ is *representable* in T if there is a formula $A(x_1, \ldots, x_n)$ such that

$$T \vdash A(\underline{a_1}, \ldots, \underline{a_n}) \quad \text{if } (a_1, \ldots, a_n) \in R,$$
$$T \vdash \neg A(\underline{a_1}, \ldots, \underline{a_n}) \quad \text{if } (a_1, \ldots, a_n) \notin R.$$

A function $f\colon \mathbb{N}^n \to \mathbb{N}$ is called *representable* in T if there is a formula $A(x_1, \ldots, x_n, y)$ representing the graph $G_f \subseteq \mathbb{N}^{n+1}$ of f, i.e., such that

$$T \vdash A(\underline{a_1}, \ldots, \underline{a_n}, \underline{f(a_1, \ldots, a_n)}), \tag{4.1}$$
$$T \vdash \neg A(\underline{a_1}, \ldots, \underline{a_n}, \underline{c}) \quad \text{if } c \neq f(a_1, \ldots, a_n) \tag{4.2}$$

and such that in addition

$$T \vdash A(\underline{a_1}, \ldots, \underline{a_n}, y) \wedge A(\underline{a_1}, \ldots, \underline{a_n}, z) \to y{=}z \text{ for all } a_1, \ldots, a_n \in \mathbb{N}. \tag{4.3}$$

Note that in case $T \vdash \underline{b} \neq \underline{c}$ for $b < c$ condition (4.2) follows from (4.1) and (4.3).

Lemma. *If the characteristic function c_R of a relation $R \subseteq \mathbb{N}^n$ is representable in T, then so is the relation R itself.*

Proof. For simplicity assume $n = 1$. Let $A(x, y)$ be a formula representing c_R. We show that $A(x, \underline{1})$ represents the relation R. Assume $a \in R$. Then $c_R(a) = 1$, hence $(a, 1) \in G_{c_R}$, hence $T \vdash A(\underline{a}, \underline{1})$. Conversely, assume $a \notin R$. Then $c_R(a) = 0$, hence $(a, 1) \notin G_{c_R}$, hence $T \vdash \neg A(\underline{a}, \underline{1})$. □

4.2.2. Undefinability of the notion of truth in formal theories.

Lemma (Fixed Point Lemma). *Assume that all elementary functions are representable in T. Then for every formula $B(z)$ we can find a closed formula A such that*

$$T \vdash A \leftrightarrow B(\underline{\ulcorner A \urcorner}).$$

PROOF. The proof is very similar to the proof of the Semantical Fixed Point Lemma. Let s be the elementary function introduced there and $A_s(x_1, x_2, x_3)$ a formula representing s in T. Let

$$C := \exists_x(B(x) \wedge A_s(z, z, x)), \quad A := C(\underline{\ulcorner C \urcorner}),$$

and therefore

$$A = \exists_x(B(x) \wedge A_s(\underline{\ulcorner C \urcorner}, \underline{\ulcorner C \urcorner}, x)).$$

Because of $s(\ulcorner C \urcorner, \ulcorner C \urcorner) = \ulcorner C(\underline{\ulcorner C \urcorner}) \urcorner = \ulcorner A \urcorner$ we can prove in T

$$A_s(\underline{\ulcorner C \urcorner}, \underline{\ulcorner C \urcorner}, x) \leftrightarrow x = \underline{\ulcorner A \urcorner},$$

hence by definition of A also

$$A \leftrightarrow \exists_x(B(x) \wedge x = \underline{\ulcorner A \urcorner})$$

and therefore

$$A \leftrightarrow B(\underline{\ulcorner A \urcorner}). \qquad \square$$

Note that for $T = \mathrm{Th}(\mathcal{M})$ we obtain the Semantical Fixed Point Lemma above as a special case.

THEOREM. *Let T be a consistent theory such that all elementary functions are representable in T. Then there cannot exist a formula $B(z)$ defining the notion of truth, i.e., such that for all closed formulas A*

$$T \vdash A \leftrightarrow B(\underline{\ulcorner A \urcorner}).$$

PROOF. Assume we would have such a $B(z)$. Consider the formula $\neg B(z)$ and choose by the Fixed Point Lemma a closed formula A such that

$$T \vdash A \leftrightarrow \neg B(\underline{\ulcorner A \urcorner}).$$

For this A we obtain $T \vdash A \leftrightarrow \neg A$, contradicting the consistency of T. $\square$

With $T := \mathrm{Th}(\mathcal{M})$ Tarski's Undefinability Theorem is a special case.

4.3. Undecidability and Incompleteness

Consider a consistent formal theory T with the property that all recursive functions are representable in T. This is a very weak assumption, as we shall show in the next section: it is always satisfied if the theory allows to develop a certain minimum of arithmetic. We shall show that such a theory necessarily is undecidable. First we shall prove a (weak) First Incompleteness Theorem saying that every axiomatized such theory must be incomplete, and then we prove a sharpened form of this theorem due to Gödel and then Rosser, which explicitly provides a closed formula A such that neither A nor $\neg A$ is provable in the theory T.

In this section let $\mathcal{L}$ again be an elementarily presented language with 0, S, $=$ in $\mathcal{L}$ and T a theory containing the equality axioms $\mathrm{Eq}_{\mathcal{L}}$.

4.3.1. Undecidability.

THEOREM (Undecidability). *Assume that T is a consistent theory such that all recursive functions are representable in T. Then T is not recursive.*

PROOF. Assume that T is recursive. By assumption there exists a formula $B(z)$ representing $\ulcorner T \urcorner$ in T. Choose by the Fixed Point Lemma a closed formula A such that

$$T \vdash A \leftrightarrow \neg B(\underline{\ulcorner A \urcorner}).$$

We shall prove $(*)$ $T \nvdash A$ and $(**)$ $T \vdash A$; this is the desired contradiction.

Ad $(*)$. Assume $T \vdash A$. Then $A \in T$, hence $\ulcorner A \urcorner \in \ulcorner T \urcorner$, hence $T \vdash B(\underline{\ulcorner A \urcorner})$ (because $B(z)$ represents in T the set $\ulcorner T \urcorner$). By the choice of A it follows that $T \vdash \neg A$, which contradicts the consistency of T.

Ad $(**)$. By $(*)$ we know $T \nvdash A$. Therefore $A \notin T$, hence $\ulcorner A \urcorner \notin \ulcorner T \urcorner$ and therefore $T \vdash \neg B(\underline{\ulcorner A \urcorner})$. By the choice of A it follows that $T \vdash A$. □

4.3.2. Incompleteness.

THEOREM (First Incompleteness Theorem). *Assume that T is an axiomatized consistent theory with the property that all recursive functions are representable in T. Then T is incomplete.*

PROOF. This is an immediate consequence of the fact that every axiomatized consistent theory which is complete is also recursive (a corollary in 4.1.3), and the Undecidability Theorem above. □

As already mentioned, we now sharpen the Incompleteness Theorem in the sense that we actually produce a formula A such that neither A nor $\neg A$ is provable. Gödel's first incompleteness theorem produced such an A under the assumption that the theory satisfied a stronger condition than mere consistency, namely "ω-consistency". Rosser then improved Gödel's result by showing, with a somewhat more complicated formula, that mere consistency is all that is required.

THEOREM (Gödel-Rosser). *Let T be axiomatized and consistent. Assume that there is a formula $L(x, y)$ – written $x < y$ – such that*

$$T \vdash \forall_{x<\underline{n}}(x = \underline{0} \vee \cdots \vee x = \underline{n-1}), \tag{4.4}$$

$$T \vdash \forall_x(x = \underline{0} \vee \cdots \vee x = \underline{n} \vee \underline{n} < x). \tag{4.5}$$

Assume also that every elementary function is representable in T. Then we can find a closed formula A such that neither A nor $\neg A$ is provable in T.

PROOF. We first define $\mathrm{Refut}_T \subseteq \mathbb{N} \times \mathbb{N}$ by

$$\mathrm{Refut}_T(d, a) := \mathrm{Prf}_T(d, \dot{\neg} a).$$

Then Refut_T is elementary and $\mathrm{Refut}_T(d,a)$ if and only if d is the GN of a derivation of the negation of a formula with GN a from a context composed of equality axioms and formulas from Ax_T. Let $B_{\mathrm{Prf}_T}(x_1,x_2)$ and $B_{\mathrm{Refut}_T}(x_1,x_2)$ be formulas representing Prf_T and Refut_T, respectively. Choose by the Fixed Point Lemma a closed formula A such that

$$T \vdash A \leftrightarrow \forall_x(B_{\mathrm{Prf}_T}(x,\underline{\ulcorner A\urcorner}) \to \exists_{y<x} B_{\mathrm{Refut}_T}(y,\underline{\ulcorner A\urcorner})).$$

A expresses its own underivability, in the form (due to Rosser): "For every proof of me there is a shorter proof of my negation".

We shall show $(*)$ $T \nvdash A$ and $(**)$ $T \nvdash \neg A$. Ad $(*)$. Assume $T \vdash A$. Choose n such that

$$\mathrm{Prf}_T(n,\ulcorner A\urcorner).$$

Then we also have

$$\text{not } \mathrm{Refut}_T(m,\ulcorner A\urcorner) \qquad \text{for all } m,$$

since T is consistent. Hence we have

$$T \vdash B_{\mathrm{Prf}_T}(\underline{n},\underline{\ulcorner A\urcorner}),$$
$$T \vdash \neg B_{\mathrm{Refut}_T}(\underline{m},\underline{\ulcorner A\urcorner}) \qquad \text{for all } m.$$

By (4.4) we can conclude

$$T \vdash B_{\mathrm{Prf}_T}(\underline{n},\underline{\ulcorner A\urcorner}) \wedge \forall_{y<\underline{n}} \neg B_{\mathrm{Refut}_T}(y,\underline{\ulcorner A\urcorner}).$$

Hence we have

$$T \vdash \exists_x(B_{\mathrm{Prf}_T}(x,\underline{\ulcorner A\urcorner}) \wedge \forall_{y<x} \neg B_{\mathrm{Refut}_T}(y,\underline{\ulcorner A\urcorner})),$$
$$T \vdash \neg A.$$

This contradicts the assumed consistency of T.

Ad $(**)$. Assume $T \vdash \neg A$. Choose n such that

$$\mathrm{Refut}_T(n,\ulcorner A\urcorner).$$

Then we also have

$$\text{not } \mathrm{Prf}_T(m,\ulcorner A\urcorner) \qquad \text{for all } m,$$

since T is consistent. Hence we have

$$T \vdash B_{\mathrm{Refut}_T}(\underline{n},\underline{\ulcorner A\urcorner}),$$
$$T \vdash \neg B_{\mathrm{Prf}_T}(\underline{m},\underline{\ulcorner A\urcorner}) \qquad \text{for all } m.$$

This implies

$$T \vdash \forall_x(B_{\mathrm{Prf}_T}(x, \underline{\ulcorner A \urcorner}) \to \exists_{y<x} B_{\mathrm{Refut}_T}(y, \underline{\ulcorner A \urcorner})),$$

as can be seen easily by cases on x, using (4.5). Hence $T \vdash A$. But this again contradicts the assumed consistency of T. □

Bibliography

E. Beth. Semantic construction of intuitionistic logic. *Medelingen de KNAW N.S.*, 19(11), 1956.

C. Chang and H. Keisler. *Model Theory*, volume 73 of *Studies in Logic*. North–Holland, Amsterdam, 3rd edition, 1990.

N. G. de Bruijn. Lambda calculus notation with nameless dummies, a tool for automatic formula manipulation, with application to the Church-Rosser theorem. *Indagationes Math.*, 34:381–392, 1972.

H.-D. Ebbinghaus, J. Flum, and W. Thomas. *Einführung in die mathematische Logik*. Spektrum Akademischer Verlag, Heidelberg, Berlin, Oxford, 4. edition, 1996.

G. Gentzen. Untersuchungen über das logische Schließen. *Mathematische Zeitschrift*, 39:176–210, 405–431, 1934.

I. Johansson. Der Minimalkalkül, ein reduzierter intuitionistischer Formalismus. *Compositio Mathematica*, 4:119–136, 1937.

L. Kalmár. Ein einfaches Beispiel für ein unentscheidbares Problem (hungarian, with german summary). *Mat. Fiz. Lapok*, 50:1–23, 1943.

A. N. Kolmogorov. Zur Deutung der intuitionistischen Logik. *Math. Zeitschr.*, 35:58–65, 1932.

J. Łoś. Quelques remarques, théorèmes et problèmes sur les classes définissables d'algèbres. In *Mathematical Interpretation of Formal Systems*, pages 98–113. North–Holland, Amsterdam, 1955.

J. Shepherdson and H. Sturgis. Computability of recursive functions. *J. Ass. Computing Machinery*, 10:217–255, 1963.

A. S. Troelstra and H. Schwichtenberg. *Basic Proof Theory*. Cambridge University Press, second edition, 2000.

Index