Mathematical Logic & Foundations of Mathematics

John Hutchinson

Mathematical Sciences Institute
Australian National University

Primary website: https://johnehutchinson.github.io
ANU website: https://maths.anu.edu.au/people/john-hutchinson

February 11, 2026

Preface

These Notes are the text for a course on Mathematical Logic at the third and fourth year honours level, or first year graduate level, for students with a strong background in pure/theoretical mathematics.

I have tried to provide a clear route through to the major results. There are various optional asides and additional material, but this is indicated as such.

Mathematical logic grew out of the work by Hilbert, Gödel, Tarski, and others in the period 1930–1950 in order to address the paradoxes and problematic foundational issues raised previously by the work of Cantor, Frege and Russell.

The subject is one which is currently taught and researched in University Departments of Mathematics, Computer Science and Philosophy, with varying emphasis on topics and approach. The emphasis here is, naturally, mathematical! But we will also consider the broader insights and applications.

Pure/theoretical mathematics is an abstract discipline, extracting, analysing, extending and axiomatising the essence of ideas that may be common to a variety of disciplines, including the physical and biological sciences, computer science, economics and increasingly the social sciences.

"Mathematical logic is the most abstract branch of mathematical thought, the most abstract human discipline.[1]

Mathematical logic abstracts further from the reasoning itself which is used in mathematics, providing insight to the limits of human and machine cognition.

Artificial Intelligence [AI]

This has an impact on the way the course is taught and assessed, and on the future of mathematical research.

The assessment is discussed in detail on the course homepage. But in essence it will be approximately 20% class participation, 20% assignments, 15% midsemester and 45% final exams. Class participation and discussion is required and is an essential part of the course.

Regarding the future of mathematics research see the Talk by Terry Tao, including the interview in the second half. There is a series of AMS articles on AI, see in particular the articles by the Australian mathematicians Terry Tao, Akshay Venkatesh and Geordie Williamson.

"Our generation is witnessing the coming-of-age of automated theorem proving and proof assistants, which seem destined to transform mathematical practice. Mathematicians of the future may be commonly using proof assistants and automated proof verifiers, with

[1]Introduction (Goldstern and Judah, 1995).

large and growing databases of formally verified proofs, to undertake their mathematical investigations.” (Hamkins, 2021, §5.4)

Contents

Introduction

In mathematics we typically prove results from a set of axioms, as for example in the study of groups, fields, rings, topology or the real number system. In mathematical logic we formalise the methods of proof, and provide axioms for the underlying logic and rules of inference

The first emphasis in these notes is on the Completeness and Incompleteness Theorems for first-order logic. This is covered in Chapters 1–4 and I have provided a direct route through to these results. This is conceptually the most challenging part of the course.

You will find the material from Chapters 5 onwards, at least to the extent it is treated here, to be more similar to the mathematics you have previously learnt.

The beginning and introductory material from the later chapters is used in Chapters 1–4. You may already know some of this material. But I always point to this material explicitly when we proceed through the first four chapters, and we will cover it in class as it is needed. I have left a more complete (and yet still very incomplete!) development of model theory, computability, the axiom of choice, ordinals, cardinals and the Zermelo–Fraenkel axioms of set theory until Chapters 5 and beyond. Most of these chapters can be expanded into a book in their own right!

Chapter 1

Structures and Languages

1.1 Truth and Proof

What is the relationship between proof and truth? Does every truth have a proof?

The following introductory comments use terminology[1] that has not yet been defined. Hopefully it will be reasonably clear from the context what is intended. Defining these concepts carefully will be a major undertaking.

Suppose for example that $\mathfrak{G}$[2] is a mathematical structure (such as a group) and $\varphi(x_1, \ldots, x_n)$ is an assertion (i.e. formula, in standard logic terminology) in some appropriate language $\mathcal{L}$ (for groups in this case). Then $\varphi(x_1, \ldots, x_n)$ will be true or false in $\mathfrak{G}$ once we assign an element a_i of $\mathfrak{G}$ to each x_i.

We will define carefully what it means for $\varphi(x_1, \ldots, x_n)$ to be *true* in $\mathfrak{G}$ for such an assignment. We write this as

$$\mathfrak{G} \vDash \varphi[a_1, \ldots, a_n] \tag{1.1}$$

and read it as "$\varphi(x_1, \ldots, x_n)$ is *true* (or *is satisfied*) in $\mathfrak{G}$ at $(a_1, \ldots, a_n)$".

Suppose now that φ is a sentence (a formula with no free variables)[3], and Σ is a set of sentences. By

$$\Sigma \vDash \varphi, \tag{1.2}$$

which we read as "Σ semantically implies φ", it is meant that φ is true in every model in which all sentences from Σ are true.

Note that (1.2) does not indicate how one might establish that $\Sigma \vDash \varphi$, just that it *is* the case for some reason or other.

To define (1.1) and (1.2) carefully we first need to define the set of all possible formulas in the language $\mathcal{L}$. For us, this set of formulas in $\mathcal{L}$ is the set of formulae in the *first-order logic*[4] for $\mathcal{L}$. We carry out that program in this Chapter.

First-order logic may seem to be too restrictive in the case of group theory since we cannot express the notion of an arbitrary subset or subgroup in first-order logic.

But all of mathematics can be developed in set theory, and set theory is most naturally developed in an appropriate first-order logic. *Much* more needs to be said about this matter, and this we will do as the course proceeds.

[1] In particular: Proof, truth, structure, language, formula, sentence.

[2] $\mathfrak{G}$ is the letter G in the Fraktur font. See Subsection 1.2.2

[3] Such as $\forall x\, \forall y\, (x \cdot y = y \cdot x)$. But not $x \cdot y = y \cdot x$, in which the variables x and y are free.

[4] "First-order" means that quantifiers $\forall$ and $\exists$ range over the elements of a structure, rather than over subsets of a structure.

The notion of truth is a semantic[5] *notion.*
The notion of proof is a syntactic[6] *notion.*

In principle one can verify and analyze a proof as a purely syntactic object, without a concept of meaning and without interpreting the language in a model.

In comparison with (1.2), with Σ and φ as there, one defines

$$\Sigma \vdash \varphi \tag{1.3}$$

to mean there is a formal proof of φ from Σ, using some deductive calculus of logical rules and axioms. Equation (1.3) is read as “there is a (formal) proof from Σ to φ”.

In Chapter 2 we develop the Hilbert proof system. We study formal proofs not with the intention of writing such proofs, but in order to understand the nature of proof and its capabilities and limits. This is currently significant in relation to proof checking and theorem proof assistant programs such as Lean.

Gödel’s Completeness Theorem in Chapter 3 is the amazing result:

$$\Sigma \vDash \varphi \Longleftrightarrow \Sigma \vdash \varphi \tag{1.4}$$

In words: The sentence φ is true whenever all the sentences (think “axioms”) in Σ are true, if and only if, there is a formal proof of φ from Σ.

The direction “$\Longleftarrow$” is straightforward, it just says that the logical rules and axioms preserve truth. But the direction “$\Longrightarrow$” is impressive: if we can deduce φ from Σ for whatever reason then there is a formal logical proof in (for example) the Hilbert system.

A Final Remark: We are interested in proofs at the meta-logic level. Hilbert systems are particularly convenient since such proofs are linear sequences, and are readily codeable arithmetically. Gentzen natural deduction systems and Gentzen sequent calculus systems, both of which are tree-like, are more appropiate for proof construction and proof analysis.

1.2 Structures

And now for the details.

A *(mathematical) structure* $\mathfrak{A}$ (e.g., a group, field, vector space, set of complex numbers, etc.) can be represented as follows:

$$\mathfrak{A} = \langle A, R_i, F_j, C_k \rangle_{\ i\in I,\ j\in J,\ k\in K} \tag{1.5}$$

where:

(i) A is the universe under consideration (e.g., the set of group elements, the set of real numbers, etc.), and $A \neq \varnothing$ for technical reasons,

[5]Semantic: adjective, relating to meaning in language or logic. Origin: mid 17th century, from French ‘sémantique’, from Greek sēmantikos ‘significant’.

[6]Syntactic: adjective, relating to the arrangement and rules for words and phrases to create well-formed sentences in a language. Origin: mid 16th century: via late Latin from Greek ‘suntaxis’, from sun ‘together’ + tassein ‘arrange’.

(ii) each R_i is an n_i-ary relation on A for some natural number $n_i \geq 1$, i.e. $R_i \subseteq A^{n_i}$,

(iii) each F_j is an n_j-ary function on A for some natural number $n_j \geq 1$, i.e. $F_j : A^{n_j} \to A$,

(iv) each C_k is a member of A, i.e. $C_k \in A$.

We shall try to be consistent in our notation. Thus the universe of a structure $\mathfrak{A}$ is A, of $\mathfrak{B}$ is B, of $\mathfrak{C}$ is C, etc. (See the following NOTES regarding the Fraktur Font used here.)

1.2.1 EXAMPLES

1. The structure for a group with multiplication $\circ$ and identity element e might be written $\mathfrak{G} = \langle G, \circ, e\rangle$.

We could include a unary inverse operator denoted by the symbol ι or $^{-1}$, so the structure is then $\mathfrak{G} = \langle G, \circ, \iota, e\rangle$ or $\mathfrak{G} = \langle G, \circ, ^{-1}, e\rangle$. Alternatively we could treat the inverse as a defined operator via the axioms for group theory. There is considerable flexibility and we will not need to be concerned with this from a foundational perspective.

2. The structure for a linear ordering is of the form $\mathfrak{L} = \langle L, <\rangle$, where L is a set and $<$ is a linear ordering of L.

3. The structure for a simple undirected graph is $\mathfrak{G} = \langle V, E\rangle$, where V is the set of vertices and E is a binary relation interpreted by $E(v, w)$ iff there is an edge between v and w.

4. Zermelo–Frankel set theory is discussed in Chapter 13. The language is particularly simple as it has just a single binary relation symbol, interpreted as membership and usually written $\in$. If we want to distinguish between the symbol and its interpretation in a structure, we occasionally write E for the symbol.

What makes this impressive is that all of mathematics can be formulated in set theory. What makes this strange is that there are countable models of the axioms, by the Lowenheim–Skolem Theorem ref ***. We discuss this paradox later ref ***

5. The (first-order) structure for the real number system is $\mathfrak{R} = \langle \mathbb{R}, +, \cdot, <, 0, 1, \rangle$ where $+$ and $\cdot$ are binary operators on $\mathbb{R}$, $<$ is a binary relation, 0 and 1 are constants, all with their standard interpretation.

However, the completeness axiom for the real number system refers to arbitrary subsets of $\mathbb{R}$ and so is not a first-order concept. On the other hand, we can imbed $\mathfrak{R}$ within set theory, which *is* first-order. Again, we discuss later ref ***

1.2.2 NOTES

1. *Fraktur Font*: For structures/models in mathematical logic and for Lie algebras, the Fraktur Font is standard usage. In LaTeX use `\mathfrak{ }`.

Here it is:

$$\mathfrak{A}, \mathfrak{B}, \mathfrak{C}, \mathfrak{D}, \mathfrak{E}, \mathfrak{F}, \mathfrak{G}, \mathfrak{H}, \mathfrak{I}, \mathfrak{J}, \mathfrak{K}, \mathfrak{L}, \mathfrak{M}, \mathfrak{N}, \mathfrak{O}, \mathfrak{P}, \mathfrak{Q}, \mathfrak{R}, \mathfrak{S}, \mathfrak{T}, \mathfrak{U}, \mathfrak{V}, \mathfrak{W}, \mathfrak{X}, \mathfrak{Y}, \mathfrak{Z}.$$
$$\mathfrak{a}, \mathfrak{b}, \mathfrak{c}, \mathfrak{d}, \mathfrak{e}, \mathfrak{f}, \mathfrak{g}, \mathfrak{h}, \mathfrak{i}, \mathfrak{j}, \mathfrak{k}, \mathfrak{l}, \mathfrak{m}, \mathfrak{n}, \mathfrak{o}, \mathfrak{p}, \mathfrak{q}, \mathfrak{r}, \mathfrak{s}, \mathfrak{t}, \mathfrak{u}, \mathfrak{v}, \mathfrak{w}, \mathfrak{x}, \mathfrak{y}, \mathfrak{z}.$$

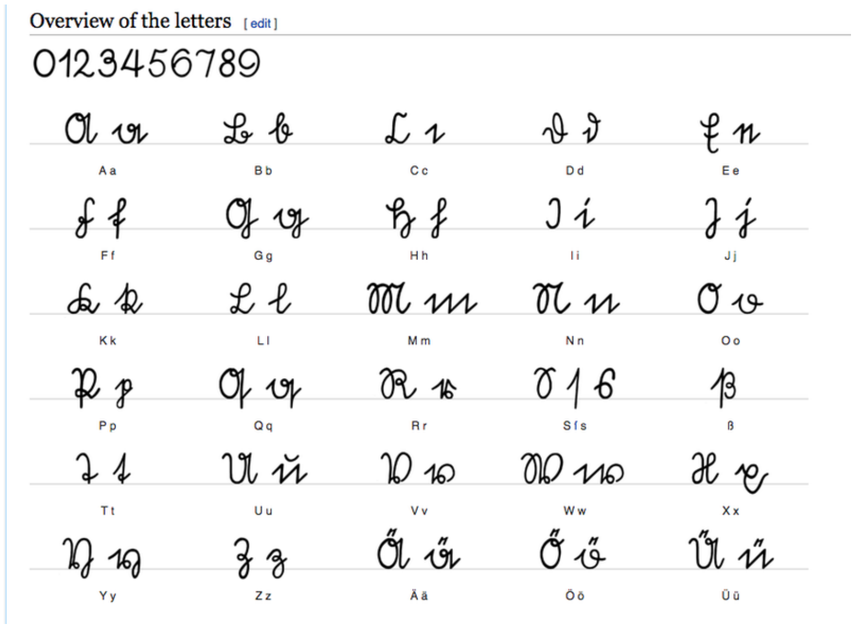


Figure 1.1: Handwritten version of the Fraktur font.

The script version in Figure 1.1 is for handwritten notes, blackboard/whiteboard talk and discussion, or to impress your friends.

Only the capitals $\mathfrak{A}$, $\mathfrak{B}$ and $\mathfrak{C}$ are commonly used in mathematical logic, while $\mathfrak{G}$, $\mathfrak{M}$, $\mathfrak{N}$ and $\mathfrak{L}$ are occasionaly used. This should not be a major stumbling block in your learning process.

2. *Greek Alphabet*: Used extensively.

α (alpha), β (beta), γ (gamma), δ (delta), ϵ (epsilon), ε (varepsilon), ζ (zeta),
η (eta), θ (theta), ϑ (vartheta), ι (iota), κ (kappa), λ (lambda), μ (mu), ν (nu),
ξ (xi), o (omicron), π (pi), ϖ (varpi), ρ (rho), ϱ (varrho), σ (sigma), ς (varsigma),
τ (tau), υ (upsilon), φ (phi), φ (varphi), χ (chi), ψ (psi), ω (omega).

and

Γ (Gamma), Δ (Delta), Θ (Theta), Λ (Lambda), Ξ (Xi), Π (Pi), Σ (Sigma),
Υ (Upsilon), Φ (Phi), Ψ (Psi), Ω (Omega).

Omicron is typeset the same as the roman letter o, and the missing capitals are typeset the same as their roman equivalents.

3. *Nonempty Universe*: The restriction that $A \neq \varnothing$ is standard in mathematical logic, although not universal in mathematical practice. See (Johnstone, 1987, page 24) for the price one must pay in terms of a modified version of modus ponens, if one allows an empty universe.

4. *Many-Sorted Structure*: In the case of a vector space, the universe could be the (disjoint) union of the set of scalars and the set of vectors. Unary relations S, V could be used to single out these two sets/sorts.

5. *Inadvisable Alternatives*: An n-ary function can be identified with an $(n+1)$-ary relation in the natural way.

A structure in which there are only constants and relations is called a *relational structure*. One could restrict considerations to relational structures, with essentially no loss of generality, as it does make some details easier.

Constants are also particular (unary) relations, or 0-ary functions. But it is inconvenient to work without constants.

6. *Foundational matters*: In certain questions of a constructive and foundational nature we restrict I, J, K to be finite. The languages, formulas, etc. can then be coded up algorithmically by integers. We say more about this as we proceed.

Another useful situation is to restrict I and J to be finite while allowing the index K for constant symbols to be infinite (for example, having one constant symbol for *every* element in some non-finite structure). See Chapter 3.

The generalisation of results to higher cardinalities usually involves no significant new ideas, other than using ordinals[7] for enumeration/iteration purposes.

1.3 Language and Syntax

We see in this Section how the symbols of a language $\mathcal{L}$ are used to construct terms, formulas, sentences, etc.

Syntax refers to the manner in which such meaningful expressions are built up in first-order logic, and to the notion and properties of formal proofs.

1.3.1 Symbols of a Language

A *language* $\mathcal{L}$ consists of a set of symbols. The language to describe the previous structure $\mathfrak{A}$ in (1.5) consists of:

- a set of relation symbols $\{r_i : i \in I\}$, each of some assigned arity,
- a set of function symbols $\{f_j : j \in J\}$, each of some assigned arity,
- a set of constant symbols $\{c_k : k \in K\}$.

1.3.2 Structures for a Language

The structure $\mathfrak{A} = \langle A, R_i, F_j, C_k\rangle_{\,i\in I,\ j\in J,\ k\in K}$ in (1.5) is an $\mathcal{L}$-*structure* if there is a one-one correspondence $r_i \leftrightarrow R_i$, $f_j \leftrightarrow F_j$ and $c_k \leftrightarrow C_k$, such that the arities match up.

The interpretation in $\mathfrak{A}$ of a relation symbol $r \in \mathcal{L}$ is often written $r^{\mathfrak{A}}$. Similarly the interpretation of a function symbol f is often written $f^{\mathfrak{A}}$. And the interpretation of a constant symbol c is often written $c^{\mathfrak{A}}$.

So we will often write instead of (1.5),

$$\mathfrak{A} = \langle A, r_i^{\mathfrak{A}}, f_j^{\mathfrak{A}}, c_k^{\mathfrak{A}}\rangle_{\,i\in I, j\in J, k\in K} \quad \text{or just} \quad \mathfrak{A} = \langle A, r^{\mathfrak{A}}, f^{\mathfrak{A}}, c^{\mathfrak{A}}\rangle, \tag{1.6}$$

if it is clear from context what is intended.

[7]See Chapter 11.

Remark 1.1.

1. *Important Distiction*: The relation/function/constant *symbols* in a formal language $\mathcal{L}$, and their *interpretation* as actual relations/functions/constants in a structure $\mathfrak{A}$, should not be confused with each other.

2. *Different Structures – Same Language*: Think of groups and the language of group theory.

   Even the *Peano Axioms* for arithmetic have infinitely many non-isomorphic models apart from the standard model. Their study can often tell us something significant about the standard model. See Chapter 8.

3. *Uncluttering*: Eventually we may just rely on context to distinguish between symbols in a language and their interpretations in a structure, and use the same notation in both cases.

   Some authors do not like this. But always making a distinction via notation between symbol and interpretation can lead to very cluttered writing. □

1.3.3 Logical Symbols

Besides the relation, function and constant symbols specific to the language $\mathcal{L}$, we also have the following *logical symbols* common to all first-order languages, and which are used to build up the terms and formulas of $\mathcal{L}$:

(,)	parentheses,
$v_0, v_1, v_2, \ldots$	variable symbols (usually just called variables),
$\neg$	not,
$\wedge$	and,
$\forall$	for all,
$=$	equals.

The intention, as we explain subsequently, is that variables will be interpreted as elements of the universe A of some structure $\mathfrak{A}$ under consideration.

Instead of taking $v_0, v_1, \ldots, v_n, \ldots$ as variables, we could take $v, v', v'', v''', \ldots$ if we wish to make do with a finite number of symbols.

We normally use $x, y, u, v, \ldots$ to denote variables.

Definition 1.2. The set of variables for a language $\mathcal{L}$ is denoted by $\mathcal{V}$.

The symbols "$\neg$" and "$\wedge$" are *connectives*, "$\forall$" is a *quantifier*, "$=$" is a binary relation symbol (which will always be interpreted in a structure with the meaning "is the same as", rather than as a more general equivalence relation).

We take these logical symbols as *primitive* and define other logic symbols from them, see Subsection 1.3.5.

1.3.4 MetaLanguage and MetaLanguage Symbols

We have a mathematical/formal language $\mathcal{L}$ to describe our mathematical structures $\mathfrak{A}$. But we also have to discuss $\mathcal{L}$ itself and its relationship with structures $\mathfrak{A}$, and that we do in our *metalanguage*, which is English.

Unless we are doing concrete foundational matters the symbols of $\mathcal{L}$ are often treated as set-theoretic objects. This is particularly the case if we have an infinite number of symbols in $\mathcal{L}$ — as is the case when we are considering infinite structures and we add a new constant symbol for each element of the structure. See the proof of Gödel's Completeness Theorem 3.17.

It will be convenient to use the following symbols in our metalanguage: $\Longrightarrow$ or $\Rightarrow$ for "implies", $\Longleftrightarrow$ for "if and only if", & for "and".

Also, $A := B$ is an abbreviation for "A by definition equals B", $A =: B$ is an abbreviation for "B by definition equals A".

The symbol "=" is used in a number of different ways. It is a logical symbol in the language $\mathcal{L}$.

But it is also used in a metalinguistic manner, when we might write for example that $\varphi = \varphi_1 \wedge \varphi_2$ or $\varphi = \varphi(x_1, \ldots, x_n)$. This would indicate (perhaps depending on the context) that the formula φ is the conjunction of two formulas, or is a formula whose free variable are among $x_1, \ldots, x_n$.

1.3.5 Defined Logical Symbols

Propositional Logic

In the following I will assume a little knowledge on your part of propositional logic. If you know what is a truth table and what is a tautology, you should be fine. If not, please read and understand Section 5.1.

Definition 1.3. Taking $\neg, \wedge, \forall$ as primitive, the following definitions/abbreviations are justified in terms of truth tables as in Section 5.1, and the standard meaning of $\exists$.

- (*or*): $\varphi \vee \psi$ for $\neg(\neg\varphi \wedge \neg\psi)$
- (*implies*): $\varphi \rightarrow \psi$ for $\neg(\varphi \wedge \neg\psi)$
- (*if and only if*): $\varphi \leftrightarrow \psi$ for $(\varphi \rightarrow \psi) \wedge (\psi \rightarrow \varphi)$
- (*there exists*): $\exists v\, \varphi$ for $\neg\forall v \neg \varphi$

Brackets are dropped where possible, and sometimes added for clarity.

The convention is that $\neg$ is less binding than $\wedge$ and $\vee$, which are less binding than $\rightarrow$ and $\leftrightarrow$.

For example, $\neg\varphi \vee \psi \rightarrow \theta \wedge \varphi$ means $\big((\neg\varphi) \vee \psi\big) \rightarrow (\theta \wedge \varphi)$.

1.3.6 Terms

A finite sequence (or string) of symbols is a *term* of $\mathcal{L}$ if and only if it is one of the following forms:

(i) A variable v or constant symbol c;

(ii) $f(t_1, \ldots, t_n)$, where f is an n-ary function symbol of $\mathcal{L}$, and each t_i is a term.

Alternatively, we can define terms set-theoretically:

The set of terms of $\mathcal{L}$ is the least set S of finite sequences such that:

(i) each variable and constant symbol belongs to S;

(ii) if f is an n-ary function symbol of $\mathcal{L}$ and $t_1, \dots, t_n \in S$, then $f(t_1, \dots, t_n) \in S$.

Intuitively a *term* is a variable or a constant symbol, or is built from them by using function symbols. A term has a value in an $\mathcal{L}$-structure once the variables in the term are assigned values in the structure.

1.3.7 Atomic Formulas

A finite sequence (or string) of symbols is an *atomic formula* of $\mathcal{L}$ iff it is one of the following forms:

(i) $t_1 = t_2$, where t_1 and t_2 are terms;

(ii) $r(t_1, \dots, t_n)$, where r is an n-ary relation symbol and the t_i are terms.

Intuitively an *atomic formula* is the simplest expression that will be either true of false in a $\mathcal{L}$-structure once values are assigned to the variables in the formula.

1.3.8 Formulas

A finite sequence of symbols is a *formula* of $\mathcal{L}$ iff it is one of the following forms:

(i) an atomic formula;

(ii) $(\neg\varphi)$, where φ is a formula;

(iii) $(\varphi_1 \wedge \varphi_2)$, $(\varphi_1 \vee \varphi_2)$, $(\varphi_1 \to \varphi_2)$, $(\varphi_1 \leftrightarrow \varphi_2)$, where φ_1 and φ_2 are formulas;

(iv) $\forall v\, \varphi$, $\exists v\, \varphi$, where v is a variable and φ is a formula.

As for terms, we can define the sets of atomic formulas and of formulas set theoretically.

Intuitively a *formula* is an expression that is either true or false in a $\mathcal{L}$-structure once values are assigned to the free[8] variables in the formula.

Keep the following distinction in mind. The difference between $\mathcal{L}$-terms and $\mathcal{L}$-formulas is the following: once values in an $\mathcal{L}$-structure $\mathfrak{A}$ are assigned to (free) variables in a term or a formula, the term will represent an *element in* $\mathfrak{A}$ and the formula will be a *statement about* $\mathfrak{A}$ which will be either true or false.

See Subsections 1.4.1 and 1.4.2.

[8] See Subsection 1.3.15.

1.3.9 Complexity of Terms and Formulas

It is convenient to use a small number of logical symbols when proving results about formulas.

For this reason we will normally just prove results for formulas built up using the logical symbols $\neg, \wedge, \forall$, and regard $(\varphi_1 \vee \varphi_2)$, $(\varphi_1 \to \varphi_2)$, $(\varphi_1 \leftrightarrow \varphi_2)$ and $\exists v\, \varphi$ as convenient abbreviations.

It is convenient to assign an integer $h(t)$ to terms t, and $h(\varphi)$ to formulas φ, where h corresponds to the maximum number of "steps" required in their construction. If you think of the construction as being given by a finite tree, then h is the height/complexity of this tree. See Figure 1.2.

For terms define the *term complexity* function h by:

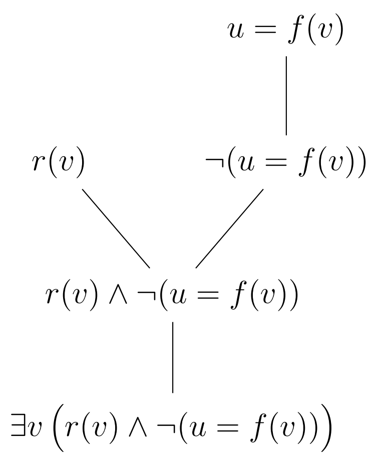
- $h(v) = h(c) = 1$, where v is a variable symbol and c is a constant symbol.[9]
- $h\big(f(t_1, \ldots, t_n)\big) = 1 + \max\{h(t_1), \ldots, h(t_n)\}$, where $f(t_1, \ldots, t_n)$ is a term.

Similarly for formulas φ, define a (different) *formula height/complexity* function h by:

- $h(\varphi) = 1$ if φ is atomic,
- $h(\neg\varphi) = 1 + h(\varphi)$,
- $h(\varphi \wedge \psi) = 1 + \max\{h(\varphi), h(\psi)\}$,
- $h(\forall v\, \varphi) = 1 + h(\varphi)$.

Here we have taken $\neg, \wedge, \forall$ as primitive symbols.

See Figure 1.2.


$u = f(v)$
$r(v)$
$\neg(u = f(v))$
$r(v) \wedge \neg(u = f(v))$
$\exists v\,\big(r(v) \wedge \neg(u = f(v))\big)$

In the diagram the atomic formulas $r(v)$ and $u = f(v)$ both have (formula) complexity 1.

The formula $\neg(u = f(v))$ has complexity 2.

The formula $\exists v\,(r(v) \wedge \neg(u = f(v)))$ has complexity 4, corresponding to the 4 levels required for its construction.

The terms $v, u, f(v)$ have (term) complexity $1, 1, 2$ respectively.

Figure 1.2: Decomposition of a first-order formula into subformulas.

Exercise 1.4. What is the complexity of $r(v) \wedge \neg(u = f(v))$?

[9] Notice that "$=$" as used here is equality in the metalanguage. It is of course *not* the equality "$=$" for the formal language $\mathcal{L}$.

Some authors use a different symbol such as "$\equiv$" within the formal language. I think ambiguity and relying on context leads to a cleaner and less-cluttered exposition, but be careful!

1.3.10 Proving Results about Terms and Formulas

Structural properties of terms and formulas can be established by induction on their complexity. To show every formula has some property P, we must establish four things:

1. Every atomic formula has property P;
2. If ψ is $\neg\varphi$ and φ has property P, then so does ψ;
3. If ψ is $\varphi_1 \wedge \varphi_2$ and both φ_1 and φ_2 have property P, then so does ψ;
4. If ψ is $\forall v\, \varphi$ and φ has property P, then so does ψ.

1.3.11 Uniqueness of Parsing

If terms or formulas are regarded as *concatenated strings* of symbols, uniqueness of parsing becomes an issue. Could a string of symbols be represented, say, as both $(\varphi_1 \wedge \varphi_2)$ and $(\varphi_1' \wedge \varphi_2')$, etc.? The answer is no, by a suitably induction on their complexity.

We will often drop, or sometimes add, brackets for the sake of readability and according to standard mathematical practice.

If we are interested in formulas primarily as concatenated strings, it is preferable to write $\wedge\varphi_1\varphi_2$ instead of $(\varphi_1 \wedge \varphi_2)$, $\neg\varphi$ instead of $(\neg\varphi\ )$, $\forall v\varphi$ instead of $(\forall v)\varphi$. This is known as *Polish notation.* We won't bother about doing this.

1.3.12 Other Logics

The language $\mathcal{L}$, the notions of term and formula of $\mathcal{L}$, the interpretation of these terms and formulas in structures for $\mathcal{L}$ (Section 1.4), and provability in $\mathcal{L}$ (Chapter 2), together determine what we refer to as the *first-order logic* over $\mathcal{L}$. Terminology is not universally consistent — often "language" and "logic" are used interchangeably.

Various other logics are possible over $\mathcal{L}$. If we add a second class of variables, interpreted as ranging over *subsets* of the interpretation, we obtain *second-order logic.* Likewise, higher-order logics.

Alternatively, new logical symbols can be added, such as a quantifier Q where $Qv\ldots$ is interpreted as meaning "there are uncountably many v such that ...", etc. Or, we may allow conjunctions and disjunctions of sets of formulas of certain infinite cardinalities. Also, various combinations of these ideas.

These and other logics have been studied extensively. Although they are not nearly as well-behaved as first-order logic, some of them do have nice properties.

From the perspective of studying the foundations of mathematics, first-order logic is the most important.

Moreover, all of mathematics can be done within set theory, and set theory has a first-order theory, as we discuss in Chapter 13.

1.3.13 Cardinality of a Language

The *cardinality* of a language $\mathcal{L}$ is

$$|\mathcal{L}| := \max\{\aleph_0, |I|, |J|, |K|\}, \tag{1.7}$$

where $|I|, |J|, |K|$ are the cardinalities of the sets I, J, K of relation, function and constant symbols respectively of $\mathcal{L}$. See (1.5), (1.6)

Exercise 1.5. $|\mathcal{L}|$ is the cardinality of the set of formulas of $\mathcal{L}$.

1.3.14 Subformulas

The set of *subformulas* of a given formula is defined inductively. In particular, the set of subformulas:

- of an atomic formula just contains the formula itself,
- of $\neg\varphi$ is $\{\neg\varphi\} \cup \{\text{subformulas of } \varphi\}$,
- of $\varphi \wedge \psi$ is $\{\varphi \wedge \psi\} \cup \{\text{subformulas of } \varphi\} \cup \{\text{subformulas of } \psi\}$,
- of $\forall v\, \varphi$ is $\{\forall v\, \varphi\} \cup \{\text{subformulas of } \varphi\}$.

And similarly for other logical symbols.

Exercise 1.6. What is the set of subformulas of $r(v) \wedge \neg(u = f(v))$? See Figure 1.2. $\square$

1.3.15 Free and Bound Variables

Intuitively, the *free* variables of a formula φ are those variables that occur in φ but are not within the scope of a quantifier. They are the variables on whose values, in a given structure, the truth or falsity of φ depends (see Section 1.4).

The *bound* variables are those that occur within the scope of a quantifier.

Definition 1.7. The set $FV(\varphi)$ of *free variables* of a formula φ is defined inductively:

- If φ is an atomic formula then $FV(\varphi)$ is the set of all variables that occur in φ.
- $FV(\neg\varphi) = FV(\varphi)$.
- $FV(\varphi \wedge \psi) = FV(\varphi) \cup FV(\psi)$.
- $FV(\forall v\, \varphi) = FV(\varphi) \setminus \{v\}$.

The symbol "\" denotes set-theoretic subtraction.

Regarding defined logical symbols, it follows that

- $FV(\varphi \vee \psi) = FV(\varphi \rightarrow \psi) = FV(\varphi \leftrightarrow \psi) = FV(\varphi) \cup FV(\psi)$.
- $FV(\exists v\, \varphi) = FV(\varphi) \setminus \{v\}$.

The set $BV(\varphi)$ of *bound variables* of φ is similarly defined by induction on complexity, and is the set of variables of φ occurring within the scope of a quantifier.

Exercise 1.8. Look at the following examples. Then write out the definition for $BV(\varphi)$. □

Example 1.9 *(Free and Bound variables).*

$\varphi_1 := r(u,v) \to u = w$	$FV(\varphi_1) = \{u,v,w\}$	$BV(\varphi_1) = \varnothing$
$\varphi_2 := \exists v \big(r(u,v) \to u = w\big)$	$FV(\varphi_2) = \{u,w\}$	$BV(\varphi_2) = \{v\}$
$\varphi_3 := \forall u\, \exists v \big(r(u,v) \to u = w\big)$	$FV(\varphi_3) = \{w\}$	$BV(\varphi_3) = \{u,v\}$
$\varphi_4 := \forall u\, \exists v \big(r(u,v) \to u = w\big) \vee (u = u)$	$FV(\varphi_4) = \{u,w\}$	$BV(\varphi_4) = \{u,v\}$
$\varphi_5 := \forall x\, \exists v \big(r(x,v) \to x = w\big) \vee (u = u)$	$FV(\varphi_5) = \{u,w\}$	$BV(\varphi_5) = \{x,v\}$

It is often more convenient to speak of *free and bound occurrences* of a variable. In particular, for φ_4 as compared to φ_3, there are 2 additional occurrences of the symbol "u", both of which are free. The first 3 occurences of u in both cases are bound. The last two occurrences of u in φ_4 are free occurrences.

It is also conventional to say in φ_2 that the 2 occurrences of the symbol v are bound, in φ_3 the 3 occurrences of u are bound, etc.

Finally, note that φ_4 and φ_5 both have the same intended meaning. In fact, it is always possible and often convenient, to change the bound variables without changing any intended meaning. In this way one can ensure that no variable symbol in a formula has both free and bound occurrences.

1.3.16 Sentences

A *sentence* is a formula with no free variables.

So a sentence is always true or false in a structure, and this is independent of any assignment of values to variables.

1.3.17 Notation Regarding Free Variables

The expression $t(x_1, \ldots, x_n)$ is often used to denote a term t whose variables are a subset of $\{x_1, \ldots, x_n\}$.

But sometimes we will instead write, for example $t(v)$ or $t(u,v)$ etc., to denote a term t where u is *only one* of its free variables, or u and v are *only two* of its free variables, etc.

This will be convenient when we make substitutions. For example, if in the term $t(u)$ we replace all variables u by w, we might write $t(w)$ for the new term. It would be understood from context in this case that the free occurrences of u in t have been replaced throughout by w.

Similarly as for terms, the expression $\varphi(x_1, \ldots, x_n)$ denotes a formula φ whose *free* variables are a subset of $\{x_1, \ldots, x_n\}$.

Sometimes we write $\varphi(u)$ to draw attention to the fact that u is a free variable of φ, although there may also be other free variables in φ. This is again convenient if we wish to replace a free occurrence of u by another variable v or constant symbol c, in which case we write $\varphi(v)$ or $\varphi(c)$ respectively. If there is any danger of ambiguity I will be careful to explain what is intended.

1.3.18 Expansion of a Language

Sometimes we are interested in passing from a language $\mathcal{L}$ to a larger language $\mathcal{L}' \supseteq \mathcal{L}$. We say $\mathcal{L}'$ is an *expansion* of $\mathcal{L}$ and $\mathcal{L}$ is a *reduction* of $\mathcal{L}'$.

We will be particularly interested in the case where $\mathcal{L}' \setminus \mathcal{L}$ consists of new constant symbols, typically an infinite set of constant symbols, corresponding to a name for each element of some $\mathcal{L}$-structure. See the proof of Theorem 3.17.

1.4 Semantics

"Semantics" refers to the interpretation of a formal language in one or more structures.

The notion of truth of a sentence in a structure, and more generally of satisfaction of a formula for some assignment of values to its free variables, both defined in this section, are semantic notions.

Unless mentioned otherwise, we consider a fixed *language $\mathcal{L}$. All formulas, sentences, and structures are for $\mathcal{L}$.*

Suppose φ is a *sentence*, and $\mathfrak{A}$ a structure for $\mathcal{L}$. In order to define

$$\mathfrak{A} \models \varphi, \tag{1.8}$$

which we read as "φ is true in $\mathfrak{A}$" or "$\mathfrak{A}$ satisfies φ", we need to more generally consider *formulas* with free variables.

The definition of $\mathfrak{A} \models \varphi$ proceeds by an induction on the complexity of φ. For example, to decide whether $\mathfrak{A} \models \forall v\, \varphi(v)$ we need to consider the truth value of $\varphi(v)$ in $\mathfrak{A}$ for various assignments of members of A (the universe of $\mathfrak{A}$) to the variable v.

The precise notion of *truth* as defined here is due to Tarski (Tarski, 1931; 1944) While it may appear relatively natural now, as Tarski remarks there are considerable difficulties in constructing *a materially adequate and formally correct definition of the term 'true sentence'*.

1.4.1 Interpretation of a Term in a Structure

Definition 1.10 (Variable Assignment). A variable assignment σ from the set of variables $\mathcal{V} = \{v_0, v_1, \ldots, v_n, \ldots\}$ into the structure $\mathfrak{A}$ is a function

$$\sigma : \mathcal{V} \to A, \tag{1.9}$$

mapping $\mathcal{V}$ into the universe A of $\mathfrak{A}$.

The value of a term t under an assignment σ is defined in the natural way using the complexity of t.

Definition 1.11 (Interpretation of a Term). Let t be a term and let σ be a variable assignment into $\mathfrak{A}$. Then the interpretation $t^{\mathfrak{A}}[\sigma]$ of t in $\mathfrak{A}$ under the assignment σ is defined by induction on the complexity of t as follows:

(i) if t is a variable x, then $t^{\mathfrak{A}}[\sigma] = \sigma(x)$,

(ii) if t is a constant symbol c, then $t^{\mathfrak{A}}[\sigma] = c^{\mathfrak{A}}$,

(iii) if t is a term $f(t_1, \ldots, t_m)$, then

$$t^{\mathfrak{A}}[\sigma] = f^{\mathfrak{A}}\Big(t_1^{\mathfrak{A}}[\sigma], \ldots, t_m^{\mathfrak{A}}[\sigma]\Big).$$

If $t = t(x_1, \ldots, x_n)$[10] is a term all of whose variables are in the set $\{x_1, \ldots, x_n\}$, then $t^{\mathfrak{A}}[\sigma]$ depends *only* on the values $\sigma(x_1), \ldots, \sigma(x_n)$ and *not* on $\sigma(u)$ for any other variable u.

That is, if $\sigma, \sigma' : \mathcal{V} \to A$ and $\sigma(x_i) = \sigma'(x_i)$ for $i = 1, \ldots, n$, then $t^{\mathfrak{A}}[\sigma] = t^{\mathfrak{A}}[\sigma']$.

Exercise 1.12. The proof of the above assertion is a simple induction on the complexity of t. Write out the proof in a succinct manner. □

Exercise 1.13. Is the "=" in the statement $t^{\mathfrak{A}}[\sigma] = t^{\mathfrak{A}}[\sigma']$ a symbol in the language $\mathcal{L}$ or a symbol in the metalanguage? Explain. □

If $t = t(x_1, \ldots, x_n)$, and $\sigma(x_i) = a_i$ for $i = 1, \ldots, n$, then one often writes

$$t^{\mathfrak{A}}[a_1, \ldots, a_n] \quad \text{or} \quad t[a_1, \ldots, a_n]$$

for $t^{\mathfrak{A}}[\sigma]$, where $\mathfrak{A}$ is understood from context in the second case.

1.4.2 Satisfaction of a Formula in a Structure

Let $\varphi = \varphi(x_1, \ldots, x_n)$ be a formula of $\mathcal{L}$, where the set $\{x_1, \ldots, x_n\}$ contains all free variables from φ.

Let σ be a variable assignment into the $\mathcal{L}$-structure $\mathfrak{A}$, where $\sigma(x_i) = a_i$ for $i = 1, \ldots, n$. We will define

$$\mathfrak{A} \vDash \varphi[\sigma] \quad \text{or equivalently} \quad \mathfrak{A} \vDash \varphi[a_1, \ldots, a_n], \tag{1.10}$$

so that it agrees with the informal idea that $\varphi(x_1, \ldots, x_n)$ is true in $\mathfrak{A}$ if each variable $x_i \in \mathcal{V}$ is interpreted as $a_i \in A$.

Moreover, if two assignments $\sigma, \sigma' : \mathcal{V} \to A$ agree on $x_1, \ldots, x_n$, then it will follow from the definition that

$$\mathfrak{A} \vDash \varphi[\sigma] \iff \mathfrak{A} \vDash \varphi[\sigma'] \tag{1.11}$$

This justifies the use of the notation $\mathfrak{A} \vDash \varphi[a_1, \ldots, a_n]$ in (1.10).

We read (1.10) as "$\mathfrak{A}$ satisfies φ at (the assignment) σ" or "$\mathfrak{A}$ satisfies φ at $(a_1, \ldots, a_n)$", or something similar.

If φ is a sentence and so φ has no free variables, then from (1.11) the truth or falsity of $\mathfrak{A} \vDash \varphi[\sigma]$ is independent of the choice of assignment σ, and so we can simply write this as $\mathfrak{A} \vDash \varphi$.

We now give the formal definition of $\mathfrak{A} \vDash \varphi[\sigma]$ by induction on the complexity of φ. At each stage we will obtain either that $\mathfrak{A} \vDash \varphi[\sigma]$ is true or otherwise that $\mathfrak{A} \vDash \varphi[\sigma]$ is false. In the second case we write $\mathfrak{A} \nvDash \varphi[\sigma]$

In the following we are, as usual, taking $\neg, \wedge, \forall$ as the primitive logical symbols for the language $\mathcal{L}$.

[10]Once again, the "=" here is a symbol in the metalanguage, not in the formal language $\mathcal{L}$. The statement $t = t(x_1, \ldots, x_n)$ is a shorthand way of saying that the variables occurring in t belong to the set $\{x_1, \ldots, x_n\}$.

Definition 1.14 (Formula Satisfaction). Let $\mathfrak{A}$ be an $\mathcal{L}$-structure. We define for every variable assignment σ, by induction on the complexity of a formula φ, the relation $\mathfrak{A} \vDash \varphi[\sigma]$.

(i) $\mathfrak{A} \vDash (t_1 = t_2)[\sigma]$ iff $t_1^{\mathfrak{A}}[\sigma] = t_2^{\mathfrak{A}}[\sigma]$,[11]

(ii) $\mathfrak{A} \vDash r(t_1, \dots, t_m)[\sigma]$ iff $(t_1^{\mathfrak{A}}[\sigma], \dots, t_m^{\mathfrak{A}}[\sigma]) \in r^{\mathfrak{A}}$,

(iii) $\mathfrak{A} \vDash (\neg\varphi)[\sigma]$ iff $\mathfrak{A} \nvDash \varphi[\sigma]$,

(iv) $\mathfrak{A} \vDash (\varphi_1 \wedge \varphi_2)[\sigma]$ iff $\mathfrak{A} \vDash \varphi_1[\sigma]$ and $\mathfrak{A} \vDash \varphi_2[\sigma]$,

(v) $\mathfrak{A} \vDash \forall x\, \varphi[\sigma]$ iff $\mathfrak{A} \vDash \varphi[\sigma']$ for *all* σ' such that $\sigma(u) = \sigma'(u)$ whenever $u \neq x$.

It is a straightforward induction, based on the complexity of φ, to show that the truth value of $\mathfrak{A} \vDash \varphi[\sigma]$ depends only on the assignments $\sigma(x_i)$ for the *free* variables x_i in φ.

Exercise 1.15. Do this for (iii) and (v). □

1.4.3 Other Logical Symbols

We took $\neg, \wedge, \forall$ as primitive and defined $\vee, \rightarrow, \leftrightarrow, \exists$ from them. It follows from the intended meanings that

(vi) $\mathfrak{A} \vDash (\varphi_1 \vee \varphi_2)\,[\sigma]$ iff $\mathfrak{A} \vDash \varphi_1[\sigma]$ or $\mathfrak{A} \vDash \varphi_2[\sigma]$,

(vii) $\mathfrak{A} \vDash (\varphi_1 \rightarrow \varphi_2)\,[\sigma]$ iff $\mathfrak{A} \vDash \varphi_2[\sigma]$ whenever $\mathfrak{A} \vDash \varphi_1[\sigma]$,

(viii) $\mathfrak{A} \vDash (\varphi_1 \leftrightarrow \varphi_2)\,[\sigma]$ iff $\mathfrak{A} \vDash \varphi_2[\sigma]$ whenever $\mathfrak{A} \vDash \varphi_1[\sigma]$, and conversely,

(ix) $\mathfrak{A} \vDash \exists x\, \varphi[\sigma]$ iff $\mathfrak{A} \vDash \varphi[\sigma']$ for *some* σ' such that $\sigma(u) = \sigma'(u)$ whenever $u \neq x$.

Exercise 1.16. Prove (vii) and (ix). □

1.4.4 Semantic Entailment

In the following you might think of Σ as being the axioms for a group[12] or for a field.

Definition 1.17 (Semantic Entailment). Suppose Σ is a set of sentences (possibly infinite) and φ is a sentence from a language $\mathcal{L}$. Then

$$\Sigma \vDash \varphi \tag{1.12}$$

means φ is true in every structure in which Σ is true. We say Σ *semantically entails* or *logically implies* or *logically entails* φ.

If $\Sigma = \varnothing$ we write $\varnothing \vDash \varphi$ or $\vDash \varphi$, and say φ is *true in every structure*, or φ is *logically valid.*

If $\varphi = \varphi(x_1, \dots, x_n)$ is a formula with free variables included in $\{x_1, \dots, x_n\}$, then $\Sigma \vDash \varphi$ means $\Sigma \vDash \forall x_1 \dots \forall x_n\, \varphi$. □

[11]This is yet another example of using notation, here "$=$", both in the formal language as in $t_1 = t_2$, and in the metalanguage as in $t_1^{\mathfrak{A}}[\sigma] = t_2^{\mathfrak{A}}[\sigma]$.

[12]Almost all interesting results in group theory are not expressible in the first-order logic corresponding to the language of group theory. But they *are* expressible and provable in Zermelo Fraenkel set theory, which in turn *is* a first-order theory.

Inserting universal quantifiers as in the definition is common mathematical practice. For example, we might write

$$u = v \to v = u,$$

where it is clear from the context that this is intended to apply to all interpretations of u and v in the relevant structures.

Note that for a set of sentences Σ, and for sentences φ and ψ,

$$\Sigma \vDash \varphi \quad \text{iff} \quad \Sigma \cup \{\neg\varphi\} \text{ has no models.} \tag{1.13}$$

Moreover,

$$\Sigma, \psi \vDash \varphi \quad \text{iff} \quad \Sigma \vDash \psi \to \varphi. \tag{1.14}$$

The analogous, deeper *syntactic result, is the Deduction Theorem 2.22*.

1.4.5 Examples re Validity

Useful examples of logical validity and non-validity, are the following.

Exercise 1.18. Give an argument to establish this in each case.

(i) $\vDash \forall v(\varphi \to \psi) \leftrightarrow (\varphi \to \forall v\, \psi)$, v not free in φ,

(ii) $\vDash \exists v(\varphi \to \psi) \leftrightarrow (\varphi \to \exists v\, \psi)$, v not free in φ,

(iii) $\vDash \forall v(\varphi \to \psi) \leftrightarrow (\exists v\, \varphi \to \psi)$, v not free in ψ,

(iv) $\vDash \exists v(\varphi \to \psi) \leftrightarrow (\forall v\, \varphi \to \psi)$, v not free in ψ,

(v) $\vDash \exists u\, \forall v\, \varphi(u, v) \to \forall v\, \exists u\, \varphi(u, v)$,

(vi) $\nvDash \forall v\, \exists u\, \varphi(u, v) \to \exists u\, \forall v\, \varphi(u, v)$.

1.5 Further Comments, Further Reading

Chapter 2

Hilbert's Deductive System

The notions of *axioms, rules of inference and formal proof* discussed in this chapter, are *syntactic*. The notions of satisfaction and truth in a structure are *semantic*.

The axioms, the rules and the definition of proof were designed designed in such a way that it was hoped, and was shown to be true, that $\Sigma \models \varphi$ if and only if $\Sigma \vdash \varphi$. See Chapter 3.

The deductive system we use here is one of many *Hilbert*-system variants. It is mathematically simple in that proofs are linear (rather than tree-like) and so well adapted to coding as sequences of natural numbers. See Chapter 9.

But from the perspective of computer science and proof theoretic studies the approach due to Gentzen and others, in particular *natural deduction* and the *sequent calculus*, are much more useful.

Remark 2.1 *(Notation).* In the following consider a fixed language $\mathcal{L}$ unless otherwise clear from context.

Take $\neg, \wedge, \forall, =$ as primitive logical symbols and consider the symbols $\vee, \rightarrow, \leftrightarrow, \exists$, as being defined in terms of them.

We also write $u \neq v$ for $\neg(u = v)$. $\square$

2.1 Axioms

2.1.1 Preliminaries

Remark 2.2 *(Propositional Logic).* If you know a little about propositional logic, truth tables and tautologies, keep reading. If not, read and understand Subsections 5.1.1–5.1.6. $\square$

Remark 2.3 *(Logical Axioms).* The *logical axioms* for first-order logic are divided into three groups. The propositional axioms, the quantifier axioms and the equality axioms.

The intention is that the axioms should be true in every $\mathcal{L}$-structure, and that the rules of inference should preserve truth.

If an instance of an axiom has free variables then *the axiom should be true for all interpretations of the free variables in the structure.* Equivalently, its universal generalisation obtained by universally quantifying over all its free variables.

For example, if we take $u = v \rightarrow v = u$ as an axiom then we intend this to mean $\forall u\, \forall v\, (u = v \rightarrow v = u)$. $\square$

2.1.2 Propositional Axioms

Definition 2.4.

Suppose φ is a formula of $\mathcal{L}$ obtained from a *propositional tautology* ψ, by substituting simultaneously and uniformly, formulas of $\mathcal{L}$ for the propositional symbols in ψ.

Then φ is a *propositional axiom* for first-order logic.

Remark 2.5 *(Examples).* blah blah

Remark 2.6 *(Possible Variations).* We could just take as propositional axioms the substitution instances of the axioms for propositional logic in Section 5.2, rather than substitution instances for *all* tautologies.

However, for our purposes there is no need to be restrictive in this manner. The significant point is that truth tables provide an algorithmic procedure for determining if a formula in propositional logic is indeed a tautology. This can then be used to decide if a formula of $\mathcal{L}$ is a propositional axiom. $\square$

2.1.3 Quantifier Axioms

Definition 2.7.

1. Suppose φ and ψ are formulas of $\mathcal{L}$ and v *is a variable not free in* φ. Then

$$\forall v(\varphi \to \psi) \to (\varphi \to \forall v\, \psi) \tag{2.1}$$

is a *quantifier axiom*, sometimes called *distribution of* $\forall$ *over implication* (under the "variable not free" side condition on φ).

2. Suppose φ (written $\varphi(v)$ here) is a formula of $\mathcal{L}$, $\varphi(t)$ is obtained from φ by substituting the term t for each *free occurrence* of v in φ, and *no variable of* t *occurs bound in* φ *at the places where it is substituted.*[1] Then

$$\forall v\, \varphi(v) \to \varphi(t) \tag{2.2}$$

is the axiom of *universal instantiation* or of *specialisation*. In particular,

$$\forall v\, \varphi(v) \to \varphi(v). \tag{2.3}$$

(The formula $\varphi(v)$ may have other free variables besides v.)

Remark 2.8 *(Restrictions on Quantifier Axioms).* These restrictions are necessary.

(a) For example, in (2.1) suppose φ and ψ are the same formula $s(v)$, where s is a unary relation symbol in the language $\mathcal{L}$.

Then

$$\forall v\, \big(s(v) \to s(v)\big) \to \big(s(v) \to \forall v\, s(v)\big)$$

cannot be an axiom as it is not logically valid.

This is so since

$$\mathfrak{A} \vDash \forall v\, \big(s(v) \to s(v)\big)$$

[1]This is often expressed as "t is substitutable for v in φ" or "*t is free for v in φ*".

for every $\mathcal{L}$-structure $\mathfrak{A}$. But it is *not* the case in general that

$$\mathfrak{A} \models\ s(v) \to \forall v\, s(v),$$

since the *free* occurrence of v could be asssigned to some $a \in A$ such that $\mathfrak{A} \models s[a]$, while $\mathfrak{A} \not\models \forall v\, s(v)$.

(b) In (2.2) suppose φ (i.e. $\varphi(v)$) is the formula $\exists u\, r(u, v)$, where r is a binary relation symbol in the language $\mathcal{L}$..

Suppose the term t is the variable u. Then $\varphi(t)$ is $\exists u\, r(u, u)$ and (2.2) becomes

$$\forall v\, \exists u\, r(u, v) \to \exists u\, r(u, u),$$

which is *not* true in some $\mathcal{L}$-structures.

The problem is that the term t contains (and indeed is) the variable u, which become bound when substituted for v in φ.

(c) Note that in (2.2) it *is* acceptable that the term t is identical to v. So in particular, $\forall v\, \varphi \to \varphi$ (i.e. $\forall v\, \varphi(v) \to \varphi(v)$) is an axiom.

In particular, taking the example from (b) where $\varphi(v)$ is the formula $\exists u\, r(u, v)$, but now taking t to be the variable v, we get the formula

$$\forall v\, \exists u\, r(u, v) \to \exists u\, r(u, v).$$

This *is* true in all $\mathcal{L}$-structures, for all assignments of a value to the *free* occurrence of v. *Why?* □

2.1.4 Equality Axioms

Definition 2.9.

1. Suppose u, v, w are variables. Then

$$\begin{gathered} u = u, \\ u = v \to v = u, \\ (u = v \wedge v = w) \to u = w, \end{gathered} \tag{2.4}$$

are *equality axioms.*

2. Suppose $u_1, \ldots, u_n$ and $v_1, \ldots, v_n$ are variables and f is an n-ary function symbol. Then

$$\Big(u_1 = v_1 \wedge \cdots \wedge u_n = v_n\Big) \ \to\ f(u_1, \ldots, u_n) = f(v_1, \ldots, v_n) \tag{2.5}$$

is an *equality axiom.*

3. Suppose $u_1, \ldots, u_n$ and $v_1, \ldots, v_n$ are variables and r is an n-ary relation symbol. Then

$$\Big(u_1 = v_1 \wedge \cdots \wedge u_n = v_n\Big) \ \to\ \Big(r(u_1, \ldots, u_n) \to r(v_1, \ldots, v_n)\Big) \tag{2.6}$$

is an *equality axiom.*

Remark 2.10 *(Equality Axioms).* Axioms (2.4) will ensure that "=", if interpreted in a structure $\mathfrak{A} = \langle A, \dots \rangle$ as a binary relation "$\sim$", then "$\sim$" is an equivalence relation.

Axioms (2.5) and (2.6) ensure that the interpretations of the function and relation symbols in $\mathfrak{A}$ respect the equivalence classes corresponding to "$\sim$".

If there are sufficient constant symbols to name all elements of A, then factoring out to obtain $\mathfrak{A}/\!\sim := \langle A/\!\sim, \dots \rangle$ will give a structure in which "=" is intepreted as the standard identity relation. This is done in Henkin's proof of Gödel's Completeness Theorem 3.11. □

2.2 Rules of Inference

Definition 2.11.

1. (*Modus Ponens*) Suppose φ and ψ are formulas of $\mathcal{L}$, then

$$\text{from} \quad \varphi \quad \text{and} \quad \varphi \to \psi \quad \text{infer} \quad \psi. \tag{2.7}$$

2. (*Universal Generalisation*) Suppose φ is a formula of $\mathcal{L}$, then

$$\text{from} \quad \varphi \quad \text{infer} \quad \forall v\, \varphi. \tag{2.8}$$

Remark 2.12 *(Modus Ponens).* From the Latin *modus ponendo ponens* – "method that by affirming affirms". It goes back to the ancient Greeks, perhaps Theophrastus (371–287 BC).

In practice, one sometimes establishes φ and then immediately claims ψ on the basis $\varphi \to \psi$ is a propositional axiom. □

Remark 2.13 *(Universal Generalisation).* A consequence of the Generalisation Rule is that we can put universal quantifiers in front of the propositional and identity axioms.

This accords with how we should think of free variables in the axioms — the axioms should hold in any structure independently of how the free variables are interpreted.

It also accords with standard mathematical practice: if we can prove a result about a variable v without any particular assumptions on v then we can deduce $\forall v\, \varphi$. □

2.3 Syntactic Entailment

Definition 2.14 (Formal Proof). Suppose φ is a *formula* and Σ is a set of *sentences* in the language $\mathcal{L}$.

A *deduction/derivation/(formal) proof of* φ *from* Σ, or alternatively a Σ*-derivation of* φ, is a finite sequence $\varphi_1, \dots, \varphi_n$ of *formulas* such that $\varphi_n = \varphi$ and such that each φ_i is a logical axiom or a member of Σ, or deducible from earlier φ_j's by means of one of the rules of inference. □

Definition 2.15 (Syntactic Entailment). Suppose Σ is a set of sentences (possibly infinite) and φ is a formula from the language $\mathcal{L}$. Then

$$\Sigma \vdash \varphi \tag{2.9}$$

means there is a formal proof from Σ to φ. We say Σ *syntactically entails/implies* φ.

If Σ is the empty set we write $\varnothing \vdash \varphi$ or $\vdash \varphi$, and say φ is a *logical theorem.* □

Exercise 2.16. Show $\Sigma \vdash \varphi \iff \Sigma \vdash \forall v\, \varphi$.

More generally, show that $\Sigma \vdash \varphi(v_1, \ldots, v_n) \iff \Sigma \vdash \forall v_1, \ldots, \forall v_n\, \varphi(v_1, \ldots, v_n)$.

(One direction uses repeated applications of the axiom of universal instantiation (2.3). The other uses repeated applications of the rule of universal generalisation (2.8).) □

Remark 2.17 *(Comparison with Semantic Entailment).*

In the analogous Definition 1.17 for semantic entailment $\Sigma \vDash \varphi$, we required φ to be a *sentence*. But if $\varphi = \varphi(v_1, \ldots, v_n)$ is a formula we could simply define $\Sigma \vDash \varphi$ to mean $\Sigma \vDash \forall v_1 \ldots \forall v_n\, \varphi$.

In this way, Gödel's Completeness Theorem 3.11, $\Sigma \vDash \varphi \iff \Sigma \vdash \varphi$, extends from sentences φ to formulas φ. □

Remark 2.18 *(Finite Nature of Proofs).* Since derivations are finite, it is an important immediate consequence that $\Sigma \vdash \varphi$ iff $\Sigma_0 \vdash \varphi$ for some *finite* $\Sigma_0 \subseteq \Sigma$. □

Remark 2.19 *(Why these particular Axioms and Inference Rules?).* As noted previously, we are working within a Hilbert-style system.

Of course, the axioms should be true in any $\mathcal{L}$-structure and the inference rules should preserve truth. This is so, and it is called the *Soundness Theorem*.

You should think about this for each axiom (schema) and convince yourself informally that this is indeed the case.

Just about every text has its own slight variant of the axioms and rules. The main point is that the axioms and rules should be sufficient to establish all other "valid" logical formulas. This requires that they should be sufficient to prove Gödel's *Completeness Theorem*. And this is what we will show in Section 3.4. □

2.4 Properties of Derivations

We discuss just a few techniques for constructing formal derivations/proofs from the Hilbert axioms, with an emphasis on what is needed for the proof of Gödel's Completeness Theorem 3.11.

Once we have the Completeness Theorem (and the Soundness Theorem), it is usually easier to check $\Sigma \vDash \varphi$ than $\Sigma \vdash \varphi$. But of course, this is not a constructive argument for showing $\Sigma \vdash \varphi$.

Formal derivations of Hilbert type are discussed in great detail in older logic texts, see. See (Enderton, 2001; Hils and Loeser, 2019; Johnstone, 1987; Mendelson, 2015)

Contemporary developments of proof theory concentrate on different axiomatic systems, natural deduction and sequent calculus in particular.

2.4.1 An Equivalence Relation on Constant Symbols

With an eye to what will be needed in Step 4 of the proof of Gödel's Completeness Theorem 3.11, let $\mathcal{C}$ be the set of constant symbols (or some subset thereof) in the language $\mathcal{L}$. Let Σ be a set of sentences in $\mathcal{L}$. For $c_1, c_2 \in \mathcal{C}$ define

$$c_1 \sim c_2 \quad \iff \quad \Sigma \vdash c_1 = c_2. \tag{2.10}$$

The formal deduction proving *reflexivity*, $c \sim c$, is

$$\begin{array}{cl} u = u & \text{Equality axioms (2.4)} \\ \forall u\,(u = u) & \text{Generalisation (2.8)} \\ \forall u\,(u = u) \to c = c & \text{Quantifier axiom (2.2)} \\ c = c & \text{Modus Ponens (2.7)} \end{array}$$

Hence $\vdash c = c$, and in particular $\Sigma \vdash c = c$, and so $c \sim c$.

To show *symmetry*, $c \sim d \Rightarrow d \sim c$, suppose $c \sim d$. Then by definition $\Sigma \vdash c = d$.

Extend such a proof to a proof of $\Sigma \vdash d = c$ by *concatenating* at its end the following:

$$\begin{array}{rl} u = v \to v = u & \text{Equality axioms (2.4)} \\ \forall u\,\forall v\,(u = v \to v = u) & \text{Generalisation (2.8), twice} \\ (c = d \to d = c) & \text{Quantifier axiom (2.2), twice} \\ d = c & \text{Modus Ponens (2.7) and using } \Sigma \vdash c = d \end{array}$$

Hence $\Sigma \vdash d = c$, and so $d \sim c$.

Exercise 2.20. So far we have seen that $\sim$ is reflexive and symmetric. The proof of *transivity*, that $(c \sim d \;\&\; d \sim e) \Longrightarrow c \sim e$, follows a similar pattern. Do it.

2.4.2 Existential Generalisation

The conditions on φ in the following are exactly as for universal instantiation, see (2.2).

Proposition 2.21. *(Existential Generalisation) Suppose $\varphi(v)$ is a formula of $\mathcal{L}$, $\varphi(t)$ is obtained from φ by substituting t for each free occurrence of v in φ, and no variable of t occurs bound in φ at the places where it is substituted. Then*

$$\vdash \varphi(t) \to \exists v\, \varphi(v). \tag{2.11}$$

Proof. From the definition of "$\exists$" we need to establish

$$\vdash \varphi(t) \to \neg\forall v\, \neg\varphi(v).$$

As in the Remark (Modus Ponens) following Definition 2.11 it is sufficient[2] to show

$$\vdash \forall v\, \neg\varphi(v) \to \neg\varphi(t).$$

But this is an instance of universal instantiation, quantifier axiom (2.2). □

[2]In propositional logic $\vdash (P \to \neg Q) \to (Q \to \neg P)$.
So in first-order logic $\vdash \big(\forall v\, \neg\varphi(v) \to \neg\varphi(t)\big) \;\to\; \big(\varphi(t) \to \neg\forall v\, \neg\varphi(v)\big)$.

2.4.3 Deduction Theorem

*** Discuss $\Sigma = \varnothing$ case.

*** Discuss the parallel between $\vdash$ and $\rightarrow$.

*** Discuss generalisation to σ not a sentence

Comment One proves $\Sigma \vdash \sigma \rightarrow \tau$ by looking at the proof $\Sigma \cup \{\sigma\} \vdash \tau$, say $\theta_1, \ldots, \theta_n (= \tau)$ and considering how each θ_i

Theorem 2.22 (Deduction Theorem). *If Σ is a set of formulas, σ a sentence, and τ a formula, then*

$$\Sigma \cup \{\sigma\} \vdash \tau \quad \Longleftrightarrow \quad \Sigma \vdash \sigma \rightarrow \tau.$$

Proof. For the (easy) "$\Longleftarrow$" direction suppose $\Sigma \vdash \sigma \rightarrow \tau$ and let $\varphi_1, \ldots, \varphi_n (= \sigma \rightarrow \tau)$ be a corresponding proof.

Then, using modus ponens, $\varphi_1, \ldots \varphi_n, \sigma, \tau$ is a proof for $\Sigma \cup \{\sigma\} \vdash \tau$.

For the "$\Longrightarrow$" direction suppose $\Sigma \cup \{\sigma\} \vdash \tau$ and let $\varphi_1, \ldots, \varphi_n (= \tau)$ be a corresponding proof.

We show by induction on i that there is a proof of $\Sigma \vdash \sigma \rightarrow \varphi_i$.

(i) If φ_i is an axiom or $\varphi_i \in \Sigma$, then $\varphi_i,\ \varphi_i \rightarrow (\sigma \rightarrow \varphi_i),\ \sigma \rightarrow \varphi_i$ is such a proof.

(ii) If $\varphi_i = \sigma$, then $\sigma \rightarrow \sigma$ is such a proof ($\sigma \rightarrow \sigma$ is a tautology of $\mathcal{L}$).

(iii) If φ_i is deduced by modus ponens from φ_j and from $\varphi_k = \varphi_j \rightarrow \varphi_i$, $(j, k < i)$,
let $\theta_1, \ldots, \theta_n,\ \sigma \rightarrow \varphi_j$ be the proof for $\Sigma \vdash \sigma \rightarrow \varphi_j$,
and $\theta'_1, \ldots, \theta'_m,\ \sigma \rightarrow (\varphi_j \rightarrow \varphi_i)$ be the proof for $\Sigma \vdash \sigma \rightarrow (\varphi_j \rightarrow \varphi_i)$.

Then the following is a proof for $\Sigma \vdash \sigma \rightarrow \varphi_i$.

$$\begin{array}{lr}
\theta_1, \ldots, \theta_n,\ \sigma \rightarrow \varphi_j,\ \theta'_1, \ldots, \theta'_m,\ \sigma \rightarrow (\varphi_j \rightarrow \varphi_i), & \\
\big(\sigma \rightarrow (\varphi_j \rightarrow \varphi_i)\big) \rightarrow \big((\sigma \rightarrow \varphi_j) \rightarrow (\sigma \rightarrow \varphi_i)\big), & \text{(logical axiom (ii))} \\
(\sigma \rightarrow \varphi_j) \rightarrow (\sigma \rightarrow \varphi_i), & \text{(modus ponens)} \\
\sigma \rightarrow \varphi_i. & \text{(modus ponens)}
\end{array}$$

(iv) If φ_i is deduced from φ_j by generalisation, i.e. $\varphi_i = \forall v\, \varphi_j$,
let $\theta_1, \ldots, \theta_n,\ \sigma \rightarrow \varphi_j$ be a proof for $\Sigma \vdash \sigma \rightarrow \varphi_j$. Then the following is a proof for $\Sigma \vdash \sigma \rightarrow \varphi_i$.

$$\begin{array}{lr}
\theta_1, \ldots, \theta_n,\ \sigma \rightarrow \varphi_j, & \\
\forall v(\sigma \rightarrow \varphi_j), & \text{(generalisation)} \\
\forall v(\sigma \rightarrow \varphi_j) \rightarrow (\sigma \rightarrow \forall v \varphi_j), & \text{(logical axiom (iv))} \\
\sigma \rightarrow \forall\, \varphi_j. & \text{(modus ponens)}
\end{array}$$

We used the fact that σ is a sentence, not just a formula, in the second last line. $\square$

Why Must σ be a Sentence in the Deduction Theorem?

Suppose $\mathcal{L}$ has a unary relation s. Then

$$s(v) \vdash \forall v\, s(v),$$

as follows by a single application of the Universal Generalisation rule.

However, we cannot expect

$$\vdash s(v) \to \forall v\, s(v).$$

The reason is that $s(v) \to \forall v\, s(v)$ is not logically valid. There are easy examples of $\mathcal{L}$-structures $\mathfrak{A}$ in which $a \in s^{\mathfrak{A}}$ for some $a \in A$, that is $\mathfrak{A} \vDash s(v)[a]$, but $\mathfrak{A} \nvDash \forall v\, s(v)$.

We might hope that we could use Quantifier Axiom (2.1) with both φ and ψ replaced by $s(v)$, giving

$$\forall v\Big(s(v) \to s(v)\Big) \to \Big(s(v) \to \forall v\, s(v)\Big).$$

But this is not an axiom since v is free in φ, i.e. in $s(v)$.

2.5 For Later Inclusion

Non-Decidability for the Set of Valid Sentences.

Suppose $\mathcal{L}$ is finite.

It follows from the Completeness Theorem that via the axioms and rules of inference we can build an idealised machine which will eventually churn out all (and only) the valid sentences of $\mathcal{L}$. We say the set of valid sentences of $\mathcal{L}$ is **computably enumerable**. This result is quite surprising.

We can ask if an even stronger result is true: is the set of valid sentences of $\mathcal{L}$ **computably decidable**? i.e. is there a machine (algorithm) which will decide for every sentence of $\mathcal{L}$ whether or not it is valid? In the next chapter we show that the answer is **NO** (except for special $\mathcal{L}$, such as $\mathcal{L}$ consisting of only unary relation symbols). In particular, there is no algorithm which will decide of an arbitrary first-order sentence whether or not it is valid.

Alternative Hilbert Style Systems

A less common variation of the Hilbert system from that in these Notes is one where there is just the single inference rule Modus Ponens, and Universal Generalisation is a derived rule.

The disadvantage with this approach is that one needs some additional axioms. But one cannot simply add $\varphi \to \forall v\, \varphi$ as an axiom, prove φ and use modus ponens to deduce $\forall v\, \varphi$. The formula $\varphi \to \forall v\, \varphi$ is typically false if φ (i.e. $\varphi(v)$) contains v as a free variable and if the free occurrence v is interpreted arbitrarily in a structure.

On the other hand, if one can prove $\varphi(v)$ with no restrictions on v, then generalisation *does* allow one to deduce $\forall v\, \varphi$.

See Axioms for first-order logic, or (Enderton, 2001, pp117–118), or (Mendelson, 2015, Exercise 2.28 p73, Hint p426).

Chapter 3

Gödel's Completeness Theorem

3.1 Overview

Goal is to provide a system that is sound, verifiable and complete such that

$$\Sigma \vdash \varphi \iff \Sigma \models \varphi$$

> I find it incredible that we have formal proof systems for which semantic validity, a sweeping concept ranging across all possible semantic contexts, aligns exactly with provability in the system. Universal truth is thereby reduced to a finite reasoning process—every universal truth is true for a finite reason and this surely informs further philosophical questions and analysis.
>
> *David Hamkins* (Hamkins, 2021, Section 5.1)

3.2 Consistency

Remark 3.1 *(Convention).* In the following, unless otherwise clear from context, all sentences are in the same fixed language $\mathcal{L}$.

The notions of a *consistent* set of sentences and of a *maximal consistent* set of sentences are syntactic. That is, they involve formal derivations from axioms.

Definition 3.2 (Absurdum). It is sometimes convenient to have a sentence, usually denoted by "$\perp$" and called *falsum*, *absurdity* or *contradiction*, and which is false in all structures for the language $\mathcal{L}$.

We define $\perp$ by

$$\perp \;:=\; \varphi \wedge \neg\varphi$$

for any sentence φ.

The essential point is that $\perp$ is false in every $\mathcal{L}$-structure. On the other hand $\neg\perp$ is true in every $\mathcal{L}$-structure.

Definition 3.3 (Consistency). A set of sentences Σ is *consistent* if a contradiction cannot be derived from Σ. That is, if $\Sigma \nvdash \perp$.

Σ is *inconsistent* if $\Sigma \vdash \perp$.

The following gives three equivalent criteria for consistency, and the corresponding three criteria for inconsistency.

Proposition 3.4 (Consistency, Equivalent Versions). *Suppose Σ is a set of sentences in the language $\mathcal{L}$.*

Then

$$\begin{aligned} \Sigma \textit{ is consistent } &\iff \textit{for every sentence } \varphi,\ \Sigma \nvdash \varphi \textit{ or } \Sigma \nvdash \neg\varphi \\ &\iff \textit{for some sentence } \psi,\ \Sigma \nvdash \psi \\ &\iff \Sigma \nvdash \bot. \end{aligned}$$

The set Σ is inconsistent if it is not consistent. That is,

$$\begin{aligned} \Sigma \textit{ is inconsistent } &\iff \textit{for some sentence } \varphi,\ \Sigma \vdash \varphi \textit{ and } \Sigma \vdash \neg\varphi \\ &\iff \textit{for every sentence } \psi,\ \Sigma \vdash \psi \\ &\iff \Sigma \vdash \bot. \end{aligned}$$

Proof.

To prove that the three definitions of “inconsistent” are equivalent note that for any two sentences φ and ψ, $\vdash \varphi \to (\neg\varphi \to \psi)$ since it is a propositional tautology.

(a) So if for some sentence φ, $\Sigma \vdash \varphi$ and $\Sigma \vdash \neg\varphi$, it follows by juxtaposing the two deductions and using modus ponens twice that, for every ψ, $\Sigma \vdash \psi$.

(b) If $\Sigma \vdash \psi$ for every ψ, then in particular $\Sigma \vdash \bot$.

(c) If $\Sigma \vdash \bot$, since $\bot \to \varphi$ and $\bot \to \neg\varphi$ as each is a propositional axiom, it follows that $\Sigma \vdash \varphi$ and $\Sigma \vdash \neg\varphi$ (for every and hence for some φ).

The three definitions of “consistent” are also equivalent since Σ is consistent iff it is not inconsistent. But the first/second/third definitions for consistency are the negation of the first/second/third definitions for inconsistency. □

The following gives connections between consistency and derivability.

Theorem 3.5 (Properties of Consistency). *Suppose Σ is a set of sentences and σ is a sentence.*

1. *Σ is consistent iff every finite subset of Σ is consistent;*
2. *If $\Sigma \vdash \sigma$ then $\Sigma \cup \{\sigma\}$ is consistent, but not conversely.*
3. *$\Sigma \cup \{\sigma\}$ is inconsistent iff $\Sigma \vdash \neg\sigma$, $\Sigma \cup \{\sigma\}$ is consistent iff $\Sigma \nvdash \neg\sigma$;*
4. *$\Sigma \cup \{\neg\sigma\}$ is inconsistent iff $\Sigma \vdash \sigma$, $\Sigma \cup \{\neg\sigma\}$ is consistent iff $\Sigma \nvdash \sigma$.*

Proof.

1. Immediate, since proofs have finite length.
2. If $\Sigma \vdash \sigma$ then any proof from $\Sigma \cup \{\sigma\}$ (including a proof of an inconsistency) can be turned into a proof from Σ.
   (*Why* does the consistency of $\Sigma \cup \{\sigma\}$ not imply $\Sigma \vdash \sigma$?)
3. For the first statement:
   (a) If $\Sigma \cup \{\sigma\}$ is inconsistent, then $\Sigma \cup \{\sigma\} \vdash \neg\sigma$ and so $\Sigma \vdash \sigma \to \neg\sigma$ by the Deduction Theorem. But $\vdash (\sigma \to \neg\sigma) \to \neg\sigma$ as this is a propositional tautology.[1] Hence $\Sigma \vdash \neg\sigma$ by modus ponens.
   (b) If $\Sigma \vdash \neg\sigma$ then $\Sigma \cup \{\sigma\} \vdash \neg\sigma$ and $\Sigma \cup \{\sigma\} \vdash \sigma$, and so $\Sigma \cup \{\sigma\}$ is inconsistent.
   The second statement in 3 follows immediately.

[1] This follows easily by truth tables. To motivate it, note that $\sigma \to \neg\sigma$, $\neg\sigma \vee \neg\sigma$ and $\neg\sigma$ are tautologically equivalent

4. This follows from 3, using $\Sigma \vdash \sigma$ iff $\Sigma \vdash \neg\neg\sigma$. □

Definition 3.6 (Maximal Consistency). A set of sentences Σ is *maximal consistent* in the language $\mathcal{L}$ iff Σ is consistent and has no proper consistent extension in $\mathcal{L}$.

That is, Σ is consistent and $\Sigma \cup \{\sigma\}$ is consistent iff $\sigma \in \Sigma$.

Theorem 3.7 (Maximal Consistency). *A set of sentences Σ is maximal consistent iff Σ is consistent and for each sentence σ either $\sigma \in \Sigma$ or $\neg\sigma \in \Sigma$.*

Proof. First suppose Σ is maximal consistent and so in particular is consistent.

For a sentence σ, if $\sigma \in \Sigma$ we are done.

If $\sigma \notin \Sigma$ then $\Sigma \cup \{\sigma\}$ is inconsistent by maximality. So $\Sigma \vdash \neg\sigma$ by Theorem 3.5.3. Hence $\Sigma \cup \{\neg\sigma\}$ is consistent and so $\neg\sigma \in \Sigma$ by maximality.

Next suppose Σ is consistent and for each sentence σ either $\sigma \in \Sigma$ or $\neg\sigma \in \Sigma$.

If $\Sigma \vdash \sigma$ then $\Sigma \cup \{\sigma\}$ is consistent, and so $\sigma \in \Sigma$ by maximality.

If $\Sigma \nvdash \sigma$ then $\Sigma \cup \{\neg\sigma\}$ is consistent by Theorem 3.5.4, and so $\neg\sigma \in \Sigma$ by maximality. □

Corollary 3.8. *If Σ is maximal consistent then for any sentences σ, τ:*

$$\sigma \in \Sigma \iff \Sigma \vdash \sigma \tag{3.1}$$

$$\sigma \notin \Sigma \iff \neg\sigma \in \Sigma \iff \Sigma \vdash \neg\sigma \tag{3.2}$$

$$\sigma \wedge \tau \in \Sigma \iff \sigma \in \Sigma \textit{ and } \tau \in \Sigma. \tag{3.3}$$

Proof. Exercise. Keep is as brief as you can — just a few lines. □

The following result due to Lindenbaum will be applied in the proof of Gödel's Completeness Theorem in Section 3.4.

Lindenbaum's Theorem is easy to prove with the current set-up, it is essentially the same as for propositional logic and is just Theorem 5.18. But since that is in an optional section, let's prove it here.

Theorem 3.9 (Lindenbaum's Theorem). *Suppose Σ is a consistent set of sentences in the language $\mathcal{L}$. Then Σ can be extended to a maximal consistent set Σ^* in the same language $\mathcal{L}$.*

Proof. Let $\varphi_1, \varphi_2, \varphi_3, \ldots$ be an enumeration of all sentences from $\mathcal{L}$. Define

$$\begin{aligned} \Sigma_1 &= \Sigma, \\ \Sigma_{n+1} &= \begin{cases} \Sigma_n \cup \{\varphi_n\} & \text{if } \Sigma_n \cup \{\varphi_n\} \text{ is consistent,} \\ \Sigma_n & \text{otherwise,} \end{cases} \\ \Sigma^* &= \bigcup_{n \geq 1} \Sigma_n. \end{aligned} \tag{3.4}$$

Each Σ_n is consistent by construction. Hence Σ^* is consistent since derivations are finite in length.

To see that Σ^* is *maximal* consistent, suppose $\Sigma^* \cup \{\varphi\}$ is consistent. We want to show $\varphi \in \Sigma^*$.

But $\varphi = \varphi_n$ for some n, and because $\Sigma_n \cup \{\varphi_n\}$ must also be consistent since $\Sigma_n \subset \Sigma^*$, it follows by construction that $\varphi_n \in \Sigma_{n+1}$. Hence $\varphi_n \in \Sigma^*$ and so Σ^* is maximal consistent. $\square$

Remark 3.10 *("Construct" vs. "Constructivism").* The argument used in the proof of Lindenabum's theorem is called a *constructive* process in the following sense. At each stage in the construction, one of two (mutually exclusive) alternatives are available which determine whether or not to add φ_n to Σ_n. However, there is not normally an algorithm to decide which alternative is the case.[2]

While this usage of the word "construction" as here is standard mathematical practice, the proof of Lindenbaum's Theorem is not an acceptable process within the *Constructivism* view of mathematics. Similarly, the proof of Gödel's Theorem in Section 3.3 is *not* acceptable within the Constructivist program. $\square$

3.3 Gödel's Completeness Theorem – Preliminaries

In Section 3.4 we prove the following major result due to Gödel.

Theorem 3.11 (Completeness Theorem, Version 1). *Suppose Σ is a set of sentences in the language $\mathcal{L}$, φ is a sentence in $\mathcal{L}$, and $\Sigma \vDash \varphi$. Then $\Sigma \vdash \varphi$.*

Remark 3.12 *(Significance).* Suppose we know (by any means whatsoever) that φ is true whenever all sentences in Σ are true.

That is, we know φ is true in all models of Σ.

Then there is a formal first-order logic proof (in the sense previously discussed) of φ from Σ.

Although the theorem refers to first-order logic, its significance is enhanced by the fact that all of mathematics can be treated within (*first-order*) Zermelo Fraenkel set theory, perhaps with additional axioms such as the axiom of choice, the generalised continuum hypothesis, etc $\square$

Remark 3.13 *(An Important Case).* Suppose Σ has *no* models, i.e. Σ is not satisfiable.

This is the case iff $\Sigma \vDash \bot$. *Why?*

Then by Theorem 3.11, $\Sigma \vdash \bot$. That is, Σ is inconsistent. $\square$

An equivalent, and equally important version of the Completeness Theorem, is the following.

Theorem 3.14 (Completeness Theorem, Version 2). *Suppose Σ is a consistent set of sentences in the language $\mathcal{L}$. Then Σ is satisfiable (i.e. has a model).*

Proposition 3.15 (Versions 1 & 2 are Equivalent).

Proof. Version 1 implies Version 2 by the previous *Remark (An Important Case)*.

Next assume Version 2 (Theorem 3.14).

To prove Version 1 (Theorem 3.11), suppose Σ is a set of sentences, φ is a sentence, and $\Sigma \vDash \varphi$.

[2]See Chapter 7.

$$\begin{array}{lll} \therefore & \Sigma \cup \{\neg\varphi\} & \text{has no model.} \\ \therefore & \Sigma \cup \{\neg\varphi\} & \text{is not consistent by Theorem 3.14.} \\ \therefore & \Sigma \vdash \varphi & \text{by Theorem 3.5.} \end{array}$$

So Version 2 implies Version 1. □

Remark 3.16 *(Summary).* From Definition 3.4 and the Definition of satisfiability,

$$\Sigma \text{ is consistent} \iff \Sigma \nvdash \bot, \quad \Sigma \text{ is satisfiable} \iff \Sigma \nvDash \bot. \tag{3.5}$$

Theorems 3.11 and 3.14 respectively, together with the Soundness Theorem, can now be summarised as follows (using the contrapositive for Theorem 3.14):

$$\text{for all } \Sigma \text{ and } \varphi, \qquad \Sigma \vDash \varphi \iff \Sigma \vdash \varphi; \tag{3.6}$$

$$\text{for all } \Sigma, \qquad \Sigma \vDash \bot \iff \Sigma \vdash \bot. \tag{3.7}$$

□

3.4 Proof of Gödel's Completeness Theorem

Unless noted otherwise, all languages are finite or countably infinite. The ideas extend to uncountable languages, and this is important for applications to algebra. We discuss this later.

We now prove the version of Gödel's Completeness Theorem given by Theorem 3.14. Namely:

Theorem 3.17 (Gödel's Completeness Theorem). *Suppose Σ is a consistent set of sentences in the language $\mathcal{L}$. Then Σ has a model.*

Since all we have is the set Σ of consistent sentences, how do we go about building/constructing such a model $\mathfrak{A}$? In particular, what should we take for the universe A of $\mathfrak{A}$. The method here is due to Henkin and is somewhat different from that of Gödel, but is of broader applicability.

The proof is contained in the next eight pages. I have broken it down into 7 Steps

Proof

3.4.1 Step 1 Adding a Set $\mathcal{C}$ of New Constant Symbols to $\mathcal{L}$

Introduce a language $\mathcal{L}^* = \mathcal{L} \cup C$ which extends $\mathcal{L}$, by adding a countably infinite set $C = \{c_1, c_2, \dots\}$ of *new* constant symbols not in $\mathcal{L}$.

3.4.2 Step 2 Extend Σ to a theory Σ^* with $\mathcal{C}$ as a Set of Henkin Witnesses

This is the key result, due to Henkin. In the following Lemma, the constant c is called a *Henkin witness* for the formula $\exists v\, \varphi(v)$.

Lemma 3.18 (Henkin Witnesses). *There is a* consistent *extension* Σ^* *of* Σ *in the extended language* $\mathcal{L}^*$*, with the following property:*

For every *formula* $\varphi = \varphi(v)$ *of* $\mathcal{L}^*$ *(not just of* $\mathcal{L}$*) with at most one free variable depending on* φ *(here it is* v*), there is a constant* $c \in \mathcal{C}$ *(*c *depending on* φ*) such that*

$$\exists v\, \varphi(v) \to \varphi(c) \;\in\; \Sigma^*. \tag{3.8}$$

Proof. List all formulas of $\mathcal{L}^*$ (not just formulas of $\mathcal{L}$) with one free variable:

$$\varphi_1, \varphi_2, \ldots, \varphi_n, \ldots \tag{3.9}$$

Define a sequence of consistent extensions of Σ, in the language $\mathcal{L}^*$:

$$\Sigma =: \Sigma_0^* \subset \Sigma_1^* \subset \Sigma_2^* \subset \cdots \subset \Sigma_n^* \subset \ldots\ ,$$

by

$$\Sigma_n^* = \Sigma_{n-1}^* \cup \{\exists v\, \varphi_n(v) \to \varphi_n(c)\} \quad \text{for } n \geq 1, \tag{3.10}$$

where (depending on n), v is the free variable appearing in φ_n and c is the first constant symbol in C not already appearing in Σ_k^* for $k < n$. This is possible since only one new sentence is added at each stage, and so only a finite number of constant symbols from C have already been added at any (finite) stage in the process.

We claim that Σ_n^* is consistent for all n.[3]

This is true for $n = 0$ since $\Sigma_0^* = \Sigma$ and Σ is consistent.

Next assume Σ_{n-1}^* is consistent but Σ_n^* is not. Then:

$$\begin{array}{lll}
 & \Sigma_{n-1}^* \vdash \neg\big(\exists v\, \varphi_n(v) \to \varphi_n(c)\big) & \text{(Theorem 3.5.2)} \\
\therefore & \Sigma_{n-1}^* \vdash \exists v\, \varphi_n(v) \wedge \neg\varphi_n(c) & \text{(propositional logic and modus ponens)} \\
\therefore & \Sigma_{n-1}^* \vdash \exists v\, \varphi_n(v) \wedge \neg\varphi_n(w) & \text{(in previous derivation, replace } c \text{ everywhere} \\
 & & \quad \text{by a variable } w \text{ not used, free or bound)} \\
\therefore & \Sigma_{n-1}^* \vdash \exists v\, \varphi_n(v), \quad \Sigma_{n-1}^* \vdash \neg\varphi_n(w) & \text{(propositional logic and modus ponens)} \\
\therefore & \Sigma_{n-1}^* \vdash \exists v\, \varphi_n(v), \quad \Sigma_{n-1}^* \vdash \forall w\, \neg\varphi_n(w) & \text{(generalisation)} \\
\therefore & \Sigma_{n-1}^* \vdash \exists v\, \varphi_n(v), \quad \Sigma_{n-1}^* \vdash \neg\exists w\, \varphi_n(w) & \text{(definition of } \forall \text{ from } \exists)
\end{array}$$

But $\vdash \exists v\, \varphi_n(v) \to \exists w\, \varphi_n(w)$[4] and so $\Sigma_{n-1}^* \vdash \exists w\, \varphi_n(w)$ by modus ponens. This is a contradiction.

[3]This consistency is not surprising. See "Plausibility Argument" in the following boxed discussion.

[4]We need to be a little careful here. We passed from $\varphi_n(v)$ to $\varphi_n(c)$ to $\varphi_n(w)$. Moreover, c and v (as a free variable) do not occur in $\varphi_n(w)$. It follows that v is free for w in the sense of the universal instantiation axiom (2.2) and so

$$\begin{array}{ll}
\vdash \forall w \neg\varphi_n(w) \to \neg\varphi_n(v) & \text{by (2.1) as } v \text{ not free in } \varphi_n(w) \\
\vdash \forall v\big(\forall w \neg\varphi_n(w) \to \neg\varphi_n(v)\big) & \text{generalisation (2.8)} \\
\vdash \forall w \neg\varphi_n(w) \to \forall v \neg\varphi_n(v) & \text{axiom (2.2)} \\
\vdash \neg\forall v \neg\varphi_n(v) \to \neg\forall w \neg\varphi_n(w) & \text{propositional axioms} \\
\vdash \exists v\, \varphi_n(v) \to \exists w\, \varphi_n(w) & \text{definition of } \exists
\end{array}$$

Hence Σ_n^* is consistent for all n and so

$$\Sigma^* := \bigcup_{n\geq 0} \Sigma_n^* \tag{3.11}$$

is consistent.

Finally, C is a set of witnesses for Σ^*, since any formula φ from $\mathcal{L}^*$ with one free variable is φ_n for some n, and so a witnessing constant c for φ from C is introduced at the n-th stage in the previous procedure.

That is

$$\Sigma^* \vdash \exists v\, \varphi(v) \to \varphi(c), \tag{3.12}$$

where for each $\varphi \equiv \varphi(v)$, a formula with one free variable, the constant symbol c depends on the previous construction process. □

Plausibility Argument

It is natural to expect that we can consistently add $\exists v\, \varphi_n(v) \to \varphi_n(c)$ in (3.10) at the nth stage in the construction process. Here is a semantic argument.

(a) If we have a model for Σ_{n-1}^* in which $\exists v\, \varphi_n(v)$ is *true*, then interpreting the "new" symbol c as some such v, will make $\varphi_n(c)$ true and hence make the implication $\exists v\, \varphi_n(v) \to \varphi_n(c)$ true.

(b) If we have a model for Σ_{n-1}^* in which $\exists v\, \varphi_n(v)$ is *false*, then interpreting the "new" symbol c as any element in the model will make $\varphi_n(c)$ false, but the sentence $\exists v\, \varphi_n(v) \to \varphi_n(c)$ is again true. (*Why?*)

But of course we cannot argue semantically in this manner.

The correct argument is achieved by ensuring that there are enough axioms from first-order logic to justify the formal syntactic argument.

3.4.3 Step 3 Enlarge Σ^* to a Complete Theory

By Lindstrom's Theorem 3.9, the theory Σ^* in the language $\mathcal{L}^* = \mathcal{L} \cup \mathcal{C}$ has a *complete* consistent extension in the language $\mathcal{L}^*$.

Henceforth, *we will use the same notation* Σ^* *to denote this complete extension.*

As in Lindstrom's Theorem in general, *the extension is neither unique nor canonical.*

3.4.4 Step 4 An Equivalence Relation on $\mathcal{C}$

Define a relation "$\sim$" on $\mathcal{C}$ by the first "$\iff$" in the following,

$$c \sim c' \iff (c = c') \in \Sigma^* \iff \Sigma^* \vdash c = c' \iff \Sigma^* \nvdash c \neq c', \tag{3.13}$$

where the second "$\iff$" is by the maximality of Σ^*, and the last "$\iff$" is from Theorem 3.7.

Taking the negation of these statements,

$$c \nsim c' \iff (c = c') \notin \Sigma^* \iff \Sigma^* \nvdash c = c' \iff \Sigma^* \vdash c \neq c'. \tag{3.14}$$

We will show "$\sim$" is an equivalence relation[5]. The essential point in the argument is to use the equality axioms in (2.4). Details are in the following box.

[5]That is, $c \sim c$, $c \sim d \Longrightarrow d \sim c$, and $(c \sim d \;\&\; d \sim e) \Longrightarrow c \sim e$.

Definition 3.19. The equivalence class corresponding to "$\sim$" and containing c is written $[c]$.

It also follows from the construction of Σ^* that each equivalence class contains a *countably infinite* set of constant symbols from $\mathcal{C}$. *Why?*

3.4.5 Step 5 Constructing a Structure $\mathfrak{A}^*$ such that $\mathfrak{A}^* \vDash \Sigma^*$

The extension Σ^* of Σ is not canonical or uniquely determined. But once we have some such extension Σ^*, a structure $\mathfrak{A}^*$ for which $\mathfrak{A}^* \vDash \Sigma^*$ *is* uniquely determined from Σ^*, as we will see in this and the following Step.

In this Step we build the $\mathcal{L}^*$-structure $\mathfrak{A}^*$. In the next step we show $\mathfrak{A}^* \vDash \Sigma^*$.

The problem to be addressed in this Step is that the axioms for equality only ensure that the equality relation symbol "$=$" in an $\mathcal{L}^*$-structure will be interpreted as an equivalence relation, and not as the identity relation. So it is perhaps not so surprising that we will build the desired $\mathcal{L}^*$-structure by taking equivalence classes. But what will be critical is the use of the Henkin axioms from (3.8).

The universe A of $\mathfrak{A}^*$ is defined by

$$A := \{[c] : c \in \mathcal{C}\}, \tag{3.15}$$

where the relevant equivalence relation and the corresponding equivalence classes are from (3.13) and Definition 3.19.

The *interpretation* in $\mathfrak{A}^*$ of each constant symbol $c \in \mathcal{C}$ is, unsurprisingly, $[c]$.

$$c^{\mathfrak{A}^*} = [c]. \tag{3.16}$$

The *interpretation* of the remaining constant, function and relation symbols from $\mathcal{L}^* \setminus \mathcal{C} = \mathcal{L}$, are determined as follows:

1. *Constant Symbols in $\mathcal{L}$*: Let $d \in \mathcal{L}$ be a constant symbol.

We need to show there is an equivalence class $[c]$ as in (3.15) which, in a natural way, will be the interpretation of d.

For this first note that $\vdash d = d$ by the same argument as in the "Details" box showing $\vdash c = c$.

But $\vdash d = d \to \exists v\,(v = d)$ from Proposition 2.21, existential generalisation.

Also $\Sigma^* \vdash \exists v\,(v = d) \to c = d$ for some (Henkin witness) $c \in \mathcal{C}$, by Lemma 3.18.

From two applications of modus ponens, $\Sigma^* \vdash c = d$ for some $c \in \mathcal{C}$.

It follows as in the "Details" box that, if $c' \in \mathcal{C}$, then

$$\Sigma^* \vdash c' = d \quad\Longleftrightarrow\quad \Sigma^* \vdash c' = c \quad\Longleftrightarrow\quad [c] = [c'].$$

So we define

$$d^{\mathfrak{A}^*} = [c] \quad\Longleftrightarrow\quad \Sigma^* \vdash c = d, \tag{3.17}$$

and we have shown that the definition is independent of the equivalence class representative of $[c]$.

2. *Function Symbols in $\mathcal{L}$*: Let $f \in \mathcal{L}$ be an n-ary function symbol. Define the interpretation $f^{\mathfrak{A}^*} : A^n \to A$ of f in $\mathfrak{A}$ by

$$f^{\mathfrak{A}^*}([c_1], \dots, [c_n]) = [c] \quad \Longleftrightarrow \quad \Sigma^* \vdash f(c_1, \dots, c_n) = c, \tag{3.18}$$

where $c_1, \dots, c_n, c \in \mathcal{C}$.

To justify this we need to show:

(a) if $c_1, \dots, c_n \in \mathcal{C}$ then there exists $c \in \mathcal{C}$ such that $\Sigma^* \vdash f(c_1, \dots, c_n) = c$,

(b) if also $c'_1, \dots, c'_n \in \mathcal{C}$, $c_1 \sim c'_1, \dots, c_n \sim c'_n$ then $f(c_1, \dots, c_n) \sim f(c'_1, \dots, c'_n)$.

This then implies $f^{\mathfrak{A}^*} : A^n \to A$ is a total (i.e. everywhere defined) function, and that the definition of $f^{\mathfrak{A}^*}$ is independent of choice of equivalence class representatives.

The argument, similar to that for constant symbols, is as follows:

For (a) first note $\vdash f(c_1, \dots, c_n) = f(c_1, \dots, c_n)$ by a similar argument to that in the "Details" box showing $\vdash c = c$.

But $\vdash f(c_1, \dots, c_n) = f(c_1, \dots, c_n) \to \exists v\, (v = f(c_1, \dots, c_n))$ from Proposition 2.21, existential generalisation.

Also $\Sigma^* \vdash \exists v\, (v = f(c_1, \dots, c_n)) \to c = f(c_1, \dots, c_n)$ for some (Henkin witness) $c \in \mathcal{C}$, by Lemma 3.18.

From two applications of modus ponens, $\Sigma^* \vdash c = f(c_1, \dots, c_n)$ for some $c \in \mathcal{C}$.

This establishes (a).

For (b) we have

$$\begin{aligned}
&c_1 \sim c'_1, \dots, c_n \sim c'_n \\
&\qquad \Longrightarrow \quad \Sigma^* \vdash c_1 = c'_1, \dots, \Sigma^* \vdash c_n = c'_n && \text{using (3.13)} \\
&\qquad \Longrightarrow \quad \Sigma^* \vdash c_1 = c'_1 \wedge \dots \wedge c_n = c'_n && \text{basic properties of "} \vdash \text{"} \\
&\qquad \Longrightarrow \quad \Sigma^* \vdash f(c_1, \dots, c_n) = f(c'_1, \dots, c'_n) && \text{using equality axiom (2.5)} \\
&\qquad \Longrightarrow \quad f(c_1, \dots, c_n) \sim f(c'_1, \dots, c'_n) && \text{using (3.13)}
\end{aligned}$$

3. *Relation Symbols in $\mathcal{L}$*: Let $r \in \mathcal{L}$ be an n-ary relation symbol. Define the interpretation $r^{\mathfrak{A}^*} \subset A^n$ of r in $\mathfrak{A}^*$ by

$$\big([c_1], \dots, [c_n]\big) \in r^{\mathfrak{A}^*} \quad \Longleftrightarrow \quad \Sigma^* \vdash r(c_1, \dots, c_n), \tag{3.19}$$

where $c_1, \dots, c_n \in \mathcal{C}$.

To show this is well defined we need to show the definition of $r^{\mathfrak{A}^*}$ is independent of choice of equivalence class representatives. But this is by essentially the same argument as for a function symbol.

$$\begin{aligned}
&c_1 \sim c'_1, \dots, c_n \sim c'_n \\
&\quad \Longrightarrow \quad \Sigma^* \vdash c_1 = c'_1 \wedge \dots \wedge c_n = c'_n && \text{(3.13), properties of "} \vdash \text{"} \\
&\quad \Longrightarrow \quad \Sigma^* \vdash r(c_1, \dots, c_n) \leftrightarrow r(c'_1, \dots, c'_n) && \text{using equality axiom (2.6)} \\
&\quad \Longrightarrow \quad \Sigma^* \vdash r(c_1, \dots, c_n) \text{ iff } \Sigma^* \vdash r(c'_1, \dots, c'_n) && \text{properties of "} \vdash \text{"} \\
&\quad \Longrightarrow \quad \big([c_1], \dots, [c_n]\big) \in r^{\mathfrak{A}^*} \text{ iff } \big([c'_1], \dots, [c'_n]\big) \in r^{\mathfrak{A}^*} && \text{by (3.19)}
\end{aligned}$$

3.4.6 Step 6 Proof that $\mathfrak{A} \models \Sigma^*$

Because *every* element in the universe A of $\mathfrak{A}^*$ is the interpretation of a constant symbol $c \in \mathcal{C}$ (in fact many such constant symbols, but all of which are equivalent via "$\sim$"), we will not need to consider free variables when interpreting terms in $\mathfrak{A}^*$, or to consider formulas with free variables when defining the truth values of sentences in $\mathfrak{A}^*$.

Definition 3.20. Suppose t is a term in the language $\mathcal{L}^*$ (with no free variables). Then the interpretation $t^{\mathfrak{A}}$ ($\in A$) of t is defined by induction on the complexity of t as follows:

- If t is a constant symbol $c \in \mathcal{C}$ or $d \in \mathcal{L}$, then $c^{\mathfrak{A}^*}$ and $d^{\mathfrak{A}^*}$ have already been defined in (3.16) and (3.17). In particular

$$t^{\mathfrak{A}^*} = [c] \iff \Sigma^* \vdash t = c.$$

- If t is $f(t_1, \dots, t_n)$ then

$$t^{\mathfrak{A}^*} := f^{\mathfrak{A}^*}(t_1^{\mathfrak{A}^*}, \dots, t_n^{\mathfrak{A}^*}).$$

Lemma 3.21. *If t is a term in $\mathcal{L}^*$ and $c \in \mathcal{C}$, then*

$$t^{\mathfrak{A}^*} = [c] \iff \Sigma^* \vdash t = c. \tag{3.20}$$

Proof. We have already seen this for t a constant symbol.

Let t be $f(t_1, \dots, t_n)$. Suppose $t^{\mathfrak{A}^*} = [c]$.

Assume (the inductive hypothesis) that the result corresponding to (3.20) holds with t replaced by $t_1, \dots, t_n$.
Let $t_i^{\mathfrak{A}^*} = [c_i]$ and so $\Sigma^* \vdash t_i = c_i$, for $1 \le i \le n$.

Then

$$\begin{aligned}
t^{\mathfrak{A}^*} = [c] &\iff f^{\mathfrak{A}^*}(t_1^{\mathfrak{A}^*}, \dots, t_n^{\mathfrak{A}^*}) = [c] && \text{Definition 3.20} \\
&\iff f^{\mathfrak{A}^*}([c_1], \dots, [c_n]) = [c] && \text{by assumption for each } t_i^{\mathfrak{A}^*} \\
&\iff \Sigma^* \vdash f(c_1, \dots, c_n) = c && \text{by (3.18)} \\
&\iff \Sigma^* \vdash f(t_1, \dots, t_n) = c && \text{by } \Sigma^* \vdash t_i = c_i \text{ and equality axiom (2.5)} \\
&\iff \Sigma^* \vdash t = c && \text{definition of } t
\end{aligned}$$

This establishes the result. □

Theorem 3.22. *For every sentence φ in the language $\mathcal{L}^*$,*

$$\mathfrak{A}^* \models \varphi \iff \Sigma^* \vdash \varphi \iff \varphi \in \Sigma^*. \tag{3.21}$$

That is,

$$\mathfrak{A}^* \models \Sigma^*. \tag{3.22}$$

Proof. The second $\iff$ in (3.21) is by (3.1) in Corollary 3.8, since Σ^* is maximal consistent.

For the first $\iff$ it will be more convenient to work with sentences built using $\exists$ rather than $\forall$ as a primitive logical symbol. So we consider $\forall$ as a defined symbol via $\forall := \neg\exists\neg$.

The proof of the first $\iff$ will be by induction on the complexity of φ.

Moreover, unlike the situation with the definition and properties of "$\models$" for a general structure, since every element in A corresponds to a constant (in fact many) constant symbols in $\mathcal{C}$, we will be able to work just with sentences.

Atomic Sentences:

(a) $r(t_1, \dots, t_n)$: We claim

$$\mathfrak{A}^* \models r(t_1, \dots, t_n) \iff \Sigma^* \vdash r(t_1, \dots, t_n). \tag{3.23}$$

Since $t_i^{\mathfrak{A}^*} \in A$ for $1 \le i \le n$, let $t_i^{\mathfrak{A}^*} = [c_i]$.

Then

$$\begin{aligned} \mathfrak{A}^* \models r(t_1, \dots, t_n) &\iff (t_1^{\mathfrak{A}^*}, \dots, t_n^{\mathfrak{A}^*}) \in r^{\mathfrak{A}^*} && \text{Definition 1.14 (ii) for "}\models\text{"} \\ &\iff ([c_1], \dots, [c_n]) \in r^{\mathfrak{A}^*} \\ &\iff \Sigma^* \vdash r(c_1, \dots, c_n) && \text{definition in (3.19) of } r^{\mathfrak{A}^*} \end{aligned}$$

(b) $t_1 = t_2$: We claim

$$\mathfrak{A}^* \models t_1 = t_2 \iff \Sigma^* \vdash t_1 = t_2 \tag{3.24}$$

Let $t_1^{\mathfrak{A}^*} = [c_1]$, $t_2^{\mathfrak{A}^*} = [c_2]$.

Then $\Sigma^* \vdash t_1 = c_1$ and $\Sigma^* \vdash t_2 = c_2$ by Lemma 3.20.

Hence[6]

$$\begin{aligned} \mathfrak{A}^* \models t_1 = t_2 &\iff t_1^{\mathfrak{A}^*} = t_2^{\mathfrak{A}^*} && \text{Definition 1.14 (i) for } \models \\ &\iff [c_1] = [c_2] \\ &\iff c_1 \sim c_2 && \text{Definition 3.19 of } [\,\cdot\,] \text{ from } \sim \\ &\iff \Sigma^* \vdash c_1 = c_2 && \text{definition of } \sim \text{ in (3.13)} \\ &\iff \Sigma^* \vdash t_1 = t_2 && \text{see below} \end{aligned}$$

The last equivalence uses

$$\begin{aligned} &\vdash (c_1 = c_2 \wedge t_1 = c_1 \wedge t_2 = c_2) \to t_1 = t_2 \\ &\vdash (t_1 = t_2 \wedge t_1 = c_1 \wedge t_2 = c_2) \to c_1 = c_2 \end{aligned}$$

This, in turn, uses the logical axioms for equality. *Exercise*

Negation $\neg\varphi$: Assume (3.21) is true for φ. It follows that the analogue is true for $\neg\varphi$:

$$\begin{aligned} \mathfrak{A}^* \models \neg\varphi &\iff \mathfrak{A}^* \not\models \varphi \\ &\iff \Sigma^* \nvdash \varphi && \text{induction assumption} \\ &\iff \Sigma^* \vdash \neg\varphi && \text{Corollary 3.8, since } \Sigma^* \text{ is maximal consistent} \end{aligned}$$

[6]Note that the second "=" in the first line of the following display, and the "=" in the second line, are *not* equality symbols in the formal language $\mathcal{L}^*$. They are abbreviations in the metalanguage used to discuss the structure $\mathfrak{A}^*$. They are saying that $t_1^{\mathfrak{A}^*}$ and $t_2^{\mathfrak{A}^*}$ denote the same element in the universe A, and similarly that $c_1^{\mathfrak{A}^*}$ and $c_2^{\mathfrak{A}^*}$ denote the same element in A.

Conjunction $\varphi \wedge \psi$: Assume (3.21) is true for φ and ψ. It follows that the analogue is true for $\varphi \wedge \psi$:

$$\begin{array}{rcll} \mathfrak{A}^* \vDash \varphi \wedge \psi & \iff & \mathfrak{A}^* \vDash \varphi \quad \& \quad \mathfrak{A}^* \vDash \psi & \text{Definition 1.14 (iv) for } \vDash \\ & \iff & \Sigma^* \vdash \varphi \quad \& \quad \Sigma^* \vdash \psi & \text{induction assumption} \\ & \iff & \Sigma^* \vdash \varphi \wedge \psi & \text{see below} \end{array}$$

The last equivalence uses

$$\vdash \varphi \wedge \psi \quad \iff \quad \vdash \varphi \;\&\; \vdash \psi$$

For $\Rightarrow$ use the propositional axioms $\varphi \wedge \psi \to \varphi$ and $\varphi \wedge \psi \to \psi$ and modus ponens.
For $\Leftarrow$ concatenate deductions for $\vdash \varphi$ and $\vdash \psi$, follow this by the propositional axiom $\varphi \to (\psi \to \varphi \wedge \psi)$ and then two applications of modus ponens.

Existential Quantifier $\exists v\, \varphi$: Since $\exists v\, \varphi$ is a *sentence* in the language $\mathcal{L}^*$, it follows that φ has at most v as a free variable. For this reason we here write $\varphi(v)$ for φ, and by $\varphi(c)$ mean the sentence obtained by replacing all free occurrences of v in $\varphi(v)$ by c.
Then since *every* element in the universe of $\mathfrak{A}$ is $[c]$ for some $c \in \mathcal{C}$,

$$\begin{array}{rcll} \mathfrak{A}^* \vDash \exists v\, \varphi(v) & \iff & \mathfrak{A}^* \vDash \varphi(c) & \text{for some } c \in \mathcal{C}, \\ & \iff & \Sigma^* \vdash \varphi(c) & \text{by induction hypothesis, some } c \in \mathcal{C}, \\ & \iff & \Sigma^* \vdash \exists v\, \varphi(v), & \end{array}$$

where for "$\Rightarrow$" of the last $\iff$, use the first-order logic result $\vdash \varphi(c) \to \exists v\, \varphi(v)$, together with modus ponens; and for "$\Leftarrow$" use the Henkin Witness property (3.8) of $\mathcal{L}^*$, namely $\exists v\, \varphi(v) \to \varphi(c) \;\in\; \Sigma^*$, together with modus ponens.

This completes the proof of Theorem 3.21 and hence of Step 6. □

3.4.7 Step 7 Definition of $\mathfrak{A}$ & Proof that $\mathfrak{A} \vDash \Sigma$

This is now easy.
We have seen that

$$\mathfrak{A}^* \vDash \Sigma^*$$

Since $\Sigma \subset \Sigma^*$, it follows that

$$\mathfrak{A}^* \vDash \Sigma.$$

Since Σ is a set of sentences in the original language $\mathcal{L}$, Σ does not refer to the new constants $c \in \mathcal{C}$ which occur in the extended language $\mathcal{L}^* = \mathcal{L} \cup \mathcal{C}$.
We now *define* the structure $\mathfrak{A}$ to be the same as $\mathfrak{A}^*$, *except* that it is a structure just for the symbols in $\mathcal{L}$ and not for the new symbols in $\mathcal{C}$.
Since nothing else is changed, it is the case that

$$\mathfrak{A} \vDash \Sigma.$$

More precisely, we could argue by inducton on the complexity of formulas φ that $\Sigma \vDash \varphi[\sigma]$ iff $\Sigma^* \vDash \varphi[\sigma]$ for arbitrary assignments σ
This completes the proof of Gödel's Completeness Theorem 3.11. ■

An Interesting Journey

We began with a consistent set of sentences Σ in the language $\mathcal{L}$.
We then extended $\mathcal{L}$ to a language $\mathcal{L}^* = \mathcal{L} \cup \mathcal{C}$, by adding a countably infinite set $\mathcal{C}$ of new constant symbols. *Step 1*
We next extended Σ to Σ^*, a *consistent* set of sentences in $\mathcal{L}^*$, by consistently adding for *every* sentence in $\mathcal{L}^*$ of the form $\exists v\, \varphi(v)$, a Henkin sentence $\exists v\, \varphi(v) \to \varphi(c)$ for some $c \in \mathcal{C}$ depending on φ. *Step 2*
We then extended Σ^* to a *maximal consistent* set of sentences, which we also denoted by Σ^*. *Step 3*
We then defined an equivalence relation on $\mathcal{C}$ by $c \sim c'$ iff $(c = c') \in \Sigma^*$, and defined an $\mathcal{L}^*$-structure $\mathfrak{A}^*$ in which "=" was interpreted by "the same as" rather than as just an equivalence relation. This used the Henkin sentences in Σ^*. *Steps 4 and 5*
We next showed that $\mathfrak{A}^* \vDash \Sigma^*$ by again using the Henkin sentences in Σ^* and the maximal consistency of Σ^*. *Step 6*
Finally, by now *ignoring* the interpretation of the symbols in $\mathcal{C}$ we obtained the desired structure $\mathfrak{A}$ such that $\mathfrak{A} \vDash \Sigma$. *Step 7*

Alternative Approach

The approach in most texts is to iterate the production of Henkin Constants a countable number of times, take unions of the extended languages and extended sets of sentences thus obtained, and finally to iterate this iteration process itself a countable number of times. See (Johnstone, 1987; Leader, 2012; Zsák, 2025), and to a lesser extent (Enderton, 2001; Hils and Loeser, 2019; Leary and Kristiansen, 2015).

For the approach here see (Chang and Keisler, 2012) and comments in (Henkin, 1996, Observation C pp 156–157).[a]

[a](Henkin, 1996) "The Discovery of My Completeness Proof" has an interesting historical discussion, comments on the nature of mathematical discovery, and on Henkin's own indirect route to his proof of the Completeness Theorem. Observation C on pages 156,157 discusses Henkin's improvements in his proof as a result of his teaching the material. This is the essentially the proof used here in the Notes.

3.5 *Material to Include*

OLD MATERIAL

Compactness Theorem. A set of sentences Σ has a model iff every finite subset of Σ has a model.

Proof: Σ has a model iff Σ is consistent iff every finite subset of Σ is consistent iff every finite subset of Σ has a model.

Löwenheim–Skolem–Tarski Theorem. If Σ has an infinite model, then Σ has a model of every cardinality $\geq \aleph_0$.

References

The books and notes closest to the material in this course are (Leary and Kristiansen, 2015; Hils and Loeser, 2019; Johnstone, 1987; Leader, 2012; Zsák, 2025).

Of the first two books, (Leary and Kristiansen, 2015) is slower and perhaps best for a first introduction. However, in the first two chapters the excessive amount of notation and detail leads to rather tedious reading. It takes 86 pages to build up the necessary framework and prove Gödel's Completeness Theorem.

The book (Hils and Loeser, 2019) is a clean but succinct treatment and contains a lot of material — good for the second time around. It takes a mere 25 (and smaller) pages for the corresponding proof of Gödel's Completeness Theorem.

The treatment in the current notes, together with class lectures, is somewhere between these two books. Perhaps you could consider this treatment as motivation for the material in Hils and Loeser's book, or as an overview of Leary and Kristiansen's book!

*** Marker's book

(Johnstone, 1987) is the text for a Cambridge course but is not so clearly written. Chapters have no further divisions and they read more like a detailed transcription of the lectures.

The notes (Leader, 2012) and (Zsák, 2025) by subsequent lecturers for the Cambridge course are in most part closely based on Johnstone's book and are easier to follow.

The above five books/lecture notes are freely (and legally) available through the author's websites or otherwise, with links below.

Batzoglou, Serafim. 2024. *Introduction to Incompleteness: From Gödel's Theorems to Forcing and the Continuum Hypothesis*, Birkhäuser.

Lots of material. Similar approach to Chang and Keisler for Gödel's Completeness Theorem.

Chang, C. C. and H. Jerome Keisler. 2012. *Model Theory*, Dover. From 3rd ed Elsevier 1989

Well-written, the approach to Gödel's Completeness Theorem in these Notes is modelled on that in this book.

Church, Alonzo. 1956. *Introduction to Mathematical Logic*, Princeton University Press.

Primarily of historical interest.

Cunningham, Daniel W. 2023. *Mathematical Logic*, De Gruyter.

A more elementary approach with much detail.

Doxiadis, Apostolos and Christos H. Papadimitriou. 2009. *Logicomix: An Epic Search for Truth*, Bloomsbury.

A fun graphic novel in comic-book style about the life and work of Bertrand Russell and the foundations of logic.

Dudley, R. M. 2002. *Real analysis and probability*, Cambridge Studies in Advanced Mathematics, vol. 74, Cambridge University Press.

Revised reprint of the 1989 original.

Enderton, Herbert B. 2001. *A Mathematical Introduction to Logic*, 2nd ed., Academic Press.

A classic, somewhat dated.

Feferman, Anita Burdman. 1993. *From Trotsky to Gödel: The Life of Jean Van Heijenoort*, A.K. Peters.

With an Appendix by Solomon Feferman. The Fefermans knew Van Heijenoort professionally and socially.

Franzén, Torkel. 2005. *Gödel's Theorem: An Incomplete Guide to Its Use and Abuse*, A K Peters.

A well-written exposition at a mostly non-technical level.

______. 2006. *The Popular Impact of Gödel's Incompleteness Theorem*, Notices of the American Mathematical Society **53**, no. 4, 440–443.

Goldstern, Martin and Haim Judah. 1995. *The Incompleteness Phenomenon*, A K Peters. Corrected reprint, 2002.

Based on lectures given at Berkeley and Bar Ilan. Good informal exposition of various topics.

Hamkins, Joel David. 2021. *Lectures on the Philosophy of Mathematics*, MIT Press.

Henkin, Leon. 1949. *The completeness of the first-order functional calculus*, J. Symbolic Logic **14**, no. 3, 159–166.

______. 1996. *The discovery of my completeness proofs*, Bulletin of Symbolic Logic **2**, no. 2, 127–158.

Interesting historical discussion, comments on the nature of mathematical discovery, Henkin's indirect route to his proof of the Completeness Theorem. Observation C (pp. 156–157) gives improvements in the proof as a result of Henkin's course teaching, and which is the method used in these Notes.

Hils, Martin and François Loeser. 2019. *A First Journey Through Logic*, Springer.

Similar Hilbert system and completeness proof as for these notes.

Hinman, Peter G. 2005. *Fundamentals of Mathematical Logic*, A K Peters.

Mostly well-written, introductory graduate level, enough material for 4 courses at that level.

Jech, Thomas. 2003. *Set Theory: The Third Millennium Edition, Revised and Expanded*, 3rd ed., Springer.

Well written exensive treatment of modern set theory.

Johnstone, Peter T. 1987. *Notes on Logic and Set Theory*, Cambridge University Press.

Sometimes difficult to follow. No section breaks within chapters. Reads like a transcription of Johnstone's lectures at Cambridge.

Keisler, H. Jerome. 2000. *Elementary Calculus: An Infinitesimal Approach*, Dover Publications. `https://people.math.wisc.edu/~hkeisler/calc.html`.

First year calculus based on non-standard techniques.

______. 2022. *Foundations of Infinitesimal Calculus*, Prindle, Weber & Schmidt. `https://people.math.wisc.edu/~hkeisler/foundations.pdf`.

Well-written and rigorous treatment of the material in (Keisler, 2000).

Knuth, Donald E. 1976. *Mathematics and Computer Science: Coping with Finiteness*, Science **194**, no. 4271, 1235–1242.

Introduces the up-arrow $\uparrow\uparrow$ notation. Well-written, entertaining, for the non-expert.

Kozen, Dexter C. 1997. *Automata and Computability*, Undergraduate Texts in Computer Science, Springer.

For a Cornell computer science course. Discusses much of the material in these Notes.

Kunen, Kenneth. 2009. *The Foundations of Mathematics*, College Publications.

Well written, a lot of detail.

Leader, Imre. 2012. *Logic and Set Theory*.

Based on Johnstone's book, easier to follow.

Leary, Christopher C. and Lars Kristiansen. 2015. *A Friendly Introduction to Mathematical Logic*, 2nd ed., Milne Library.

Marker, David. 2015. *Metamathematics*.

For a first year graduate level course, extracts from his book.

______. 2024. *An Invitation to Mathematical Logic*, Springer.

Well written graduate level text.

Mendelson, Elliott. 2015. *Introduction to Mathematical Logic*, 6th ed., CRC Press.

A classic, but somewhat old-fashiuoned.

Meyer, Albert R. and Dennis M. Ritchie. 1967. *The complexity of loop programs*, Proceedings of the ACM National Meeting, 465–469.

The first to explicitly develop the connection between primitive/general recursive functions and for/while loops in a computer program.

Rathjen, Michael. *Proof Theory*, The Stanford Encyclopedia of Philosophy (Winter 2024 Edition).

See Section 2 for an overview discussion of Gentzen's Consistency Proof for Peano Arithmetic.

Potter, Michael. 2004. *Set Theory and Its Philosophy: A Critical Introduction*, Oxford University Press.

The interplay between mathematical results and philosophical reflection.

Rautenberg, Wolfgang. 2010. *A Concise Introduction to Mathematical Logic*, 3rd ed., Springer.

Rogers, Hartley Jr. 1987. *Theory of Recursive Functions and Effective Computability*, MIT Press.

A classic. The first four Sections in particular, are a very readable introduction.

Simpson, Stephen G. 2009. *Foundations of Mathematics*, Springer.

Smith, Peter. 2007. *An Introduction to Gödel's Theorems*, Cambridge University Press.

———. 2020. *Gödel Without (Too Many) Tears*, Cambridge University Press.

Soare, Robert I. 2016. *Turing Computability: Theory and Applications*, Springer.

An excellent readable text, with extensive motivation and history.

Srivastava, Shashi Mohan. 2013. *A Course on Mathematical Logic*, 2nd ed., Springer.

Tanaka, Kazuyuki. 2024. *Logic and Computation 1*.

Detailed slides from a lecture course.

Tarski, Alfred. 1931. *The concept of truth in formalized languages*, — Logic, Semantics, Metamathematics: Papers from 1923 to 1938 (edn 2, 1983), pp. 152–278.

A detailed analysis of truth from a philosophical and mathematical perspective.

———. 1944. *The Semantic Conception of Truth and the Foundations of Semantics*, Philosophy and Phenomenological Research **4**, no. 3, 341–376.

A readable summary of Tarski's previous work (Tarski, 1931).

Towsner, Henry. 2013. *Proof Theory Lecture Notes*.

Propositional and First-order Logic, Peano Arithmetic, Gentzen Sequent Calculus, Reverse Mathematics.

———. 2020a. *Goodstein's Theorem, Big Functions, and Unprovability*. Online Talks.

———. 2020b. *Goodstein's Theorem, ϵ_0, and Unprovability*.

Notes related to the online talks.

Uzquiano, Gabriel. 2022. *Quantifiers and Quantification*.

van Dalen, Dirk. 2013. *Logic and Structure*, 5th ed., Springer.

Wolf, Robert S. 2005. *A Tour Through Mathematical Logic*, Mathematical Association of America.

A user friendly survey.

Zach, Richard. 2016. *The Open Logic Text, available at* `https://openlogicproject.org/`.

Zsák, András. 2025. *Logic and Set Theory*.

Based on Johnstone's book, easier to follow.