GÖDEL'S INCOMPLETENESS THEOREM: A COMPUTER-BASED COURSE IN ELEMENTARY PROOF THEORY

by

WILFRED SIEG
Department of Philosophy
Columbia University

INGRID LINDSTRÖM
Department of Mathematics
University of Kansas, Lawrence

and

STEN LINDSTRÖM
Department of Philosophy
University of Uppsala

PHILOSOPHY 167 is a computer-based course in elementary proof theory at Stanford University. The main purpose of the course is to introduce (advanced) undergraduates to Gödel's incompleteness theorems (Gödel, 1931). These theorems are still, almost fifty years after their discovery, among the philosophically most striking and mathematically most important results in modern logic. They undermined a widely shared belief concerning the extent of the axiomatic method, and they refuted a particular, clearly formulated program in the foundations of mathematics (Hilbert's program). This background is sketched in section 1 of our article, where we also discuss the basic strategy of Gödel's proofs. The special problems inherent in a detailed presentation of Gödel's proofs are met in two ways:

1. the standard arguments are simplified by presenting them in a natural formal framework, the theory TEM for elementary metamathematics.
2. the arguments are furthermore given on a computer and are checked by it; this fact is exploited.

In section 2 we outline the content of the course, emphasizing the novel points involved in 1. Taking up 2, we describe the computer implementation in section 3. Finally, in section 4, we evaluate critically the curriculum,

The research reported in this article was partially supported by National Science Foundation Grants SED 77–09698, EPP 74–15106, and SED 74–15106, and Exxon Foundation Grant "Productivity Improvement in the Teaching of Philosophy" to Stanford University.

the mathematical work underlying the presentation of Gödel's proofs, and the implementation; ways of improving the course are discussed.

Development of this course has been carried out since October 1975 at the Institute for Mathematical Studies in the Social Sciences, Stanford University, under the general direction of Georg Kreisel and Patrick Suppes. The main work on developing the curriculum and the proof machinery was done in the years 1975 to 1977 by Wilfried Sieg with the collaboration of Lee Blaine and Vladimir Lifschitz. Ingrid Lindström and Sten Lindström have been responsible for curriculum development during 1977–1979.

1. PROOF THEORY AND HILBERT'S PROGRAM

Hilbert's program is briefly described as well as the impact of Gödel's incompleteness theorems on the aims of proof theory. The discussion of the strategy of Gödel's proofs will point to the peculiar difficulties for their detailed presentation.

1.1 *Hilbert's Program*

Proofs are used in mathematics to establish theorems; in proof theory, however, proofs, or rather derivations in formal systems, are the object of mathematical study. Proof theory started with Hilbert's proposal to investigate mathematical proofs that involve abstract infinitistic reasoning by finitist means. Hilbert proposed in particular his consistency program: consider a formal system T which can serve as a basis for all of classical mathematics and prove (by finitist means) the consistency of T. In this way Hilbert hoped to achieve a foundational reduction of all of mathematics to a particularly elementary part; namely, to "finitist mathematics." Though this part of mathematics had not been precisely characterized, it was clear that it was concerned with concrete spatio-temporal objects and that it was to employ only elementary combinatorial methods. When Hilbert formulated the aims of his proof theory, he had two significant facts available:

1. then-contemporary mathematics could be developed in formal systems of set or type theory.
2. the formal systems could be described in a finitist manner.

The first, purely empirical fact gave plausibility to Hilbert's belief that those systems provided a complete framework for mathematics. The second fact allowed him to formulate various properties of formal systems finitistically, in particular, the consistency property (Bernays, 1930–1931).

1.2 *Gödel's Theorems*

In 1930 Gödel published two classical results which established that Hilbert's foundational program cannot be carried out in its original form. These results are:

> *First incompleteness theorem.* If T is a formal system containing arithmetic, then there is a true sentence G which asserts its own unprovability and is such that:
> (i) If T is consistent, then G is not provable in T.
> (ii) If T is omega-consistent, then not-G is not provable in T.
>
> *Second incompleteness theorem.* If T is a consistent formal system containing arithmetic, then T does not prove CONS(T); CONS(T) is formulated in the language of T and expresses that T is consistent.

The first incompleteness theorem implies that no consistent formal theory can prove all true arithmetic sentences, thus refuting the quasiempirical completeness assumption, crucial for Hilbert's plan to settle the foundational problems of mathematics once and for all. The second incompleteness theorem shows that Hilbert's main reductive aim cannot be achieved, as it presupposes that finitist mathematics is a part of the comprehensive theory T. A consistency proof for T, however, cannot be carried out by means formalizable in T.[1] But the second theorem does not contradict Hilbert's general formalist viewpoint. Gödel remarked in his paper that

> this viewpoint presupposes only the existence of a consistency proof in which nothing but finitary means of proof are used, and it is conceivable that there exist finitary proofs that cannot be expressed in the formalism P [of *Principia Mathematica*] . . .

A modified Hilbert program has indeed been successfully pursued for arithmetic and significant parts of mathematical analysis; see Kreisel (1968) and Feferman (in press).

1.3 *The Proofs*

The further developments in proof theory are not touched upon in this course, whose core consists rather of the detailed proofs of the incompleteness theorems. These proofs exploit, quite curiously, the very facts which made Hilbert's program plausible. First, the elementary description of formal theories can be given via an *arithmetization of syntax* in purely number-theoretic terms; second, for theories T containing elementary number theory—a condition certainly satisfied by the comprehensive theories whose consistency Hilbert wanted to establish—this opens the possibility of defining the syntactic notions concerning T in T and, consequently, of reflecting on T within T. If the theorem predicate for T and the substitution relation can be defined in T, then one can show by an easy semantic argument that the sentence expressing its own unprovability is formally undecidable. Gödel gave this argument in the introduction to his paper; the semantic requirement on T is soundness: every sentence provable in T is true.

The proofs of Gödel's theorems involve more than giving nontrivial derivations in a theory T; central work has to be carried out in T's

---

[1] As a matter of fact, finitist mathematics was thought to be contained in number theory. On this assumption, even elementary number theory cannot be shown finitistically to be consistent.

metatheory, in which the syntax of T is rigorously described. One has to show, for example, that the syntactic notions can be *represented* in T. Usually this is done as follows. The syntactic objects constituting T are first coded as natural numbers and the syntactic notions, given by inductive definitions, are turned into recursive (number theoretic) predicates. Then the representability of recursive predicates (or functions) is established. The arguments are given in an informal, mathematical way. In the presentation described in section 2 one proceeds differently. The formal theory for elementary metamathematics TEM is introduced, the inductively defined syntactic notions are directly axiomatized, and their representability is formally proved in TEM. The first incompleteness theorem is obtained as a theorem of TEM and the second theorem is established by a simple metamathematical argument concerning TEM. But even this simplified presentation involves a great deal of technical work, if all details are carried out without informal shortcuts. As it happens, certain shortcuts can be systematically justified as we are working on a computer. This is explained in section 3.

2. GÖDEL'S THEOREMS FOR ZERMELO-FRAENKEL SET THEORY

In this section we outline the content of the course, sketching standard parts and elaborating on points where the presentation deviates from the usual ones. The course is divided into five parts; the first two are concerned with the informal presentation of Zermelo-Fraenkel set theory. In the third part the syntactic objects constituting the formal theory ZF are identified with binary trees, and the syntactic notions are formally given in TEM. Assuming representability and derivability conditions, self-referential sentences are constructed and used in Part 4 to establish classical theorems: Tarski's indefinability result, the first incompleteness theorem and its Rosser variant, Löb's theorem, and the second incompleteness theorem. In the final fifth part, which has not yet been implemented, the representability and derivability conditions are verified.

2.1 *Informal Metamathematics of* ZF

Part 1 starts out with a description of the cumulative hierarchy, segments of which are the intended models of ZF: the ZF-axioms just formulate principles underlying the construction of the hierarchy. Then we show that, apart from general mathematical notions like relation and function, specific number-theoretic ones can be given in set theory. Having defined the concept of a natural number explicitly (and indeed in a special syntactic form guaranteeing absoluteness), one can formulate natural representability conditions relating an informal number-theoretic notion P and its set-theoretic counterpart p; namely,

1. if $P(M1, \ldots, Mn)$, then $\mathrm{ZF} \vdash p(/M1/, \ldots, /Mn/)$,
2. if not $P(M1, \ldots, Mn)$, then $\mathrm{ZF} \vdash -p(/M1/, \ldots, /Mn/)$.

Here, $/M/$ is the von Neumann numeral for the natural number M. ZF can be replaced by ZF*, that is, ZF without the axiom of infinity. The question whether recursively defined number-theoretic functions can be represented in ZF* leads to the problem of making recursive set-theoretic definitions explicit. This problem is discussed in great detail, as it leads to a central set-theoretic fact (the recursion theorem) which is essential for the representability of recursive and inductive definitions.

In Part 2 we take up the tedious task of giving a detailed description of the syntax of ZF. We justify also the informal practice of introducing defined symbols and adding defining axioms to ZF (or ZF*) in the usual way. One associates with each formula F in the expanded language via a metamathematical translation a formula F' in the language of ZF and shows that:

1. $DE \vdash (F \leftrightarrow F')$, and
2. if $DE \vdash F$, then $T \vdash F'$

where DE is a definitional extension of T, and T is either ZF or ZF*. These considerations resolve the apparent conflict between mathematical convenience and metamathematical simplicity; one can work within definitional extensions of ZF or ZF* and yet use for metamathematical purposes the very simple description of the basic system.

2.2 *Formalization of Metamathematics in* TEM

The description of ZF in Part 2 employs informal inductive definitions; most of the metamathematical arguments proceed by induction on the syntactic notions. In the third part of the course, the formal theory TEM for elementary metamathematics is introduced. The theory is similar to Kleene's generalized arithmetic, where syntactic objects are viewed as finitely branching, inductively generated trees (Kleene, 1952). The analysis is pushed one step further here. Syntactic objects are identified with binary trees built up from the empty tree S by a pairing operation [,]. The obvious axioms are formulated in a standard first-order language.

$T1$: $[Z,Y]$ is not S
(The empty tree is not a pair.)

$T2$: if $[Z1,Z2]$ is $[Y1,Y2]$ then $Z1$ is $Y1$ and $Z2$ is $Y2$
(Two pairs are equal only if their respective components are equal.)

Ind: if $FM(S)$ and
for all Z,Y: ($FM(Z)$ and $FM(Y)$ only if FM ($[Z,Y]$))
then for all Z: $FM(Z)$
(The induction principle for arbitrary formulas FM.)

These axioms express the basic facts about syntactic objects. Their structure justifies certain kinds of very elementary inductive definitions of unary predicates, which are called S-inductive definitions.[2] Informally, the distinguishing feature of S-inductive definitions can be described as follows: syntactic objects falling under an S-inductive definition reflect directly their construction according to the generating clauses of the definition. This condition is satisfied by the finitely many syntactic notions needed for the formal presentation of ZF. But a satisfactory mathematical characterization of S-inductive definitions is still to be given. If FMi is a formula of TEM expressing the generating clauses for a predicate Ri, then TEM contains the axiom:

($T3i$) $FMi(Ri,Z)$ if and only if $Ri(Z)$.

Notice that the induction principle for Ri is provable. Given this framework, the formal description of ZF is given straightforwardly along familiar lines, and elementary metamathematical arguments can be formalized immediately.

2.3 *The Incompleteness Theorems*

In the first section of Part 4, earlier informal considerations are paralleled in TEM. Previously, number-theoretic notions were represented in ZF*; now syntactic notions are represented. For this purpose much weaker systems than ZF* are sufficient; but precisely which ones is a delicate question and has as yet no satisfactory answer. First, we have to encode syntactic objects *as sets*. From a model-theoretic point of view, this is done in a natural way; the empty set codes the empty tree, the set-theoretic pair $\langle x,y \rangle$ codes the pair $[X,Y]$, where x and y code X, respectively Y. Thus the codes are considered as binary trees and have the same structure as the syntactic objects they encode. Because the codes are built up from the empty set by pairing, they have canonical names in the language of ZF, and the coding function can be defined in TEM. (The code of an arbitrary syntactic object T is denoted by $|T|$.) The *transcription* of the generating clauses for the Ri into the language of ZF yields set-theoretic inductive definitions, which can be made explicit by the recursion theorem of Part 1. It is an easy matter to verify the appropriate representability conditions in TEM. The self-referential lemma is finally proved. Let F be an arbitrary ZF-formula with exactly one free variable; then one can construct a sentence D, such that

$$ZF^* \vdash (D \leftrightarrow F(|D|)).$$

[2] S-inductive definitions are related to the S-rudimentary attributes of Smullyan (1961).

That is, D expresses that it has property F. This lemma is the main technical tool for obtaining the various proof-theoretic results.

There are three groups of results. The first group contains Tarski's observation on the indefinability of an adequate notion of truth, the first incompleteness theorem, and Rosser's improvement. The results of the second group concern reflection principles and include Löb's theorem and Gödel's second theorem. For these latter results so-called derivability conditions have to be satisfied by the theorem predicate:

$D1$: $\text{ZF}^* \vdash \text{theo}(|X|) \rightarrow \text{theo}(|\text{theo}(|X|)|)$

$D2$: $\text{ZF}^* \vdash \text{theo}(|X \rightarrow Y|) \rightarrow (\text{theo}(|X|) \rightarrow \text{theo}(|Y|))$.

Jeroslow (1973) used literal self-referential sentences to show that $D2$, the provable closure under *modens ponens*, is redundant. That theorem and some lemmas concerning Henkin-sentences are contained in the third group of results.

3. COMPUTER IMPLEMENTATION

The online text material for the course is written in VOCAL (Smith, 1981) and is presented audiovisually using display terminals and head sets. The instructor is a computer program named EXCHECK (McDonald, 1981). In addition to presenting the text material in audiovisual form, the EXCHECK program has facilities for asking questions and checking the answers. It also offers exercises in which the student is asked to construct proofs, each line of which is checked for correctness by EXCHECK. There are no lecture sessions, but students do consult with teaching assistants. The curriculum text is also available in off-line form (Sieg & Lindström, 1978).

The *reflective* character of Gödel's proofs makes the presentation of this material and its computer implementation essentially different from that of other courses previously developed at Stanford (elementary logic and axiomatic set theory). Here, as opposed to working within a single formal system, one deals with several interrelated formal systems, primarily, the system ZF of set theory and the system TEM of elementary metamathematics. This characteristic feature of the subject matter makes it a nontrivial problem to implement it in such a way that it is possible

1. to give natural proofs which are not disturbed by an inordinate amount of tedious detail.
2. to express thoughts in each theory freely and informally.
3. to switch easily from one formal system to another.

In the first part of the course, students are asked to do proofs in ZF. They are not required to give genuinely formal derivations. But due to the nature of the proof-checking machinery the proofs have to be given in greater detail than in informal presentations of the subject. When dealing with TEM, students are similarly asked to do proofs within TEM.

The theory TEM serves as a formalized metatheory for ZF. When working in a metatheory one often has to establish that certain results are provable in the object theory. One way to do this is to axiomatize the provability relation of the object theory in the metatheory and to establish the result from these axioms. However, it is often far easier to simply derive the result directly in the object theory and then use this fact in the metatheory. For this purpose two procedures were added to the proof checker system of EXCHECK; namely ZFSTART, for starting a derivation in ZF from the metatheory, and ZFFINISH, for finishing the derivation in ZF and returning to the metatheory. Within a ZF-derivation the student may use prior results of the form $ZF \vdash F$ or $ZF^* \vdash F$, for some formula F.

4. CRITICAL EVALUATION AND FURTHER WORK

The proof theory course is still in need of further, detailed development. We close this article with a critical assessment of its present state and indicate possibilities for its improvement. Our remarks are centered around three points: the mathematical work underlying this presentation of Gödel's proofs, the organization of the curriculum, and the computer implementation.

4.1 *Mathematical Work*

The course is obviously incomplete; its last part, on the verification of representability and derivability conditions, has not yet been implemented. It is not difficult to verify the representability conditions for the notions actually used in the arguments. The verification can be carried out in TEM, and a metamathematical argument concerning TEM yields the derivability conditions. However, it would be much more efficient and elegant to show that all S-inductive definitions are representable. The problem is that there is not yet a satisfactory mathematical characterization of S-inductive definitions.

4.2 *Curriculum Organization*

As it stands, the course is difficult and requires some mathematical sophistication on the part of students; not so much for the central proof-theoretic section, but for the material on set theory. That material covers a nontrivial part of an introduction to axiomatic set theory. These observations are borne out by the reaction of the four students who worked on the course in 1977–1978 and in the spring quarter 1979. All of the students finished the material on ZF; one of them also went through the lessons on TEM, but stopped short of Part 4. Only one student completed the course.

There are obviously two ways to go. One option is to give up the original plan of making the course completely self-contained and require students to be familiar with axiomatic set theory. Then the set-theoretic material can be condensed to the bare minimum for establishing the recursion theorem. The other option is to expand the course to a two-quarter course. In this case, the material on ZF could include, for example a discussion of some strong axioms of infinity and Gödel's related views on the reasons for the incompleteness of ZF and similar systems.

4.3 *Computer Implementation*

Using the facilities provided by EXCHECK to a greater extent than has been done until now, the on-line curriculum text should be made considerably more interactive by adding further questions and thus making the student a more active participant. The on-line text also follows the off-line text too closely. It would be preferable to cover only the main line of arguments in the VOCAL lessons and expand the material by examples, discussion, and by providing motivations for the notions introduced. The VOCAL lessons should in general be easier to understand conceptually than the off-line curriculum. However, the main challenge of future work is to make more striking use of the computer; for example, by actually determining codes of self-referential statements or by carrying out nontrivial syntactic transformations.

REFERENCES

Bernays, P. Die Philosophie der Mathematik und die Hilbertsche Beweistheorie. *Blätter für Deutsche Philosophie,* 1930–1931, **4,** 326–367.

Feferman, S. (Ed.), *Theories of inductive definitions and subsystems of analysis: Recent proof-theoretical studies* (Lecture Notes in Mathematics). Berlin: Springer-Verlag, in press.

Gödel, K. Uber formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme, I. *Monatshefte für Mathematik und Physik,* 1931, **38,** 173–198.

Jeroslow, R. G. Redundancies in the Hilbert-Bernays derivability conditions for Gödel's second incompleteness theorem. *Journal of Symbolic Logic,* 1973, **38,** 359–367.

Kleene, S. C. *Introduction to metamathematics.* New York: Van Nostrand, 1952.

Kreisel, G. Survey of proof theory. *Journal of Symbolic Logic,* 1968, **33,** 321–388.

McDonald, J. The EXCHECK CAI system. In P. Suppes (Ed.), *University-level computer-assisted instruction at Stanford: 1968–1980.* Stanford, Calif.: Stanford University, Institute for Mathematical Studies in the Social Sciences, 1981.

Sieg, W., & Lindström, I. *Elementary proof theory* (Tech. Rep. 297). Stanford, Calif.: Stanford University, Institute for Mathematical Studies in the Social Sciences, 1978.

Smith, R. VOCAL: A case study in the methodology of the design of authoring languages for computer-assisted instruction. In P. Suppes (Ed.), *University-level computer-assisted instruction at Stanford: 1968-1980.* Stanford, Calif: Stanford University, Institute for Mathematical Studies in the Social Sciences, 1981.

Smullyan, R. *Theory of formal systems.* Princeton: Princeton University Press, 1961.