Annals of Communications in Mathematics
Volume 9, Number 2 (2026), 16
DOI: 10.62072/acm.2026.09032
ISSN: 2582-0818
https://www.technoskypub.com/journal/acm/

Review article

Gödel's Incompleteness Theorems Demystified: A Rigorous Formal–Metamathematical Exposition and Its Philosophical Limits

Aliefya Mahalva Ayn *

Department of Mathematics and Computer Science, Rutgers University–New Brunswick, New Brunswick, NJ, USA.

*Corresponding Author: Aliefya Mahalva Ayn. Email: ama545@scarletmail.rutgers.edu

Received: 18 May 2026; Accepted: 22 June 2026; Published: 30 June 2026

Citation: A. M. Ayn, Gödel's Incompleteness Theorems Demystified: A Rigorous Formal–Metamathematical Exposition and Its Philosophical Limits. Ann. Commun. Math. 9 **(2026)**, 16. https://doi.org/10.62072/acm.2026.09032

Abstract: This article assembles a detailed paper-level dependency chain for Gödel's First and Second Incompleteness Theorems while identifying the effectiveness, consistency, and base-theory assumptions used at each stage. It separates computable enumerability from primitive-recursive axiomatizability; develops Gödel-style prime-power coding; fixes an explicit Hilbert calculus; gives capture-avoiding substitution with separate primitive-recursive majorants depending on all relevant variable indices; and proves primitive recursiveness of witness-annotated proof verification for a merely computably enumerable axiom set. The syntactic Δ_0 and Σ_1 classes are fixed effectively, making the induction axioms of $I\Sigma_1$ primitive-recursively recognizable. The representability theorem is proved in $\mathbf{Q}$ by structural induction, using a finite-unfolding lemma, explicit Σ_1 closure under composition, and a displayed witness history for primitive recursion. For provability theory, the article defines a canonical trace-based proof predicate and gives detailed construction schemas, with explicit Σ_1-induction invariants, for proof transformations and bounded proof synthesis in $I\Sigma_1$. These yield formalized Σ_1-completeness, the Hilbert–Bernays–Löb derivability conditions, Löb's theorem, and the Second Incompleteness Theorem. The article also proves Rosser's strengthening, states Tarski's theorem as the nondefinability of $\mathrm{Th}(\mathbb{N})$ in arithmetic, and marks the limits of anti-mechanist interpretations. Its contribution is expository and organizational rather than a new incompleteness theorem.

Mathematics Subject Classification: 03F40, 03F30, 03B25, 03F45, 03A05
Keywords: incompleteness; arithmetization; representability; diagonal lemma; Rosser sentence; Löb's theorem; Tarski's theorem; foundations of mathematics

1 Introduction

At the beginning of the twentieth century, foundational research sought precise axiomatic systems in which mathematical proofs could be mechanically checked and whose consistency could be established by methods regarded as secure. Russell's paradox had shown that unrestricted set formation was untenable, while *Principia Mathematica*, axiomatic set theory, and Hilbert's proof-theoretic program offered disciplined

alternatives. Hilbert and Bernays presented the mature formalist program as an investigation of formal theories by finitistically acceptable metamathematics [8].

Gödel's 1931 paper transformed this program. It showed that a consistent, effectively axiomatized theory containing enough arithmetic cannot decide every arithmetical sentence and, under the standard arithmetization of provability, cannot establish its own consistency by means formalizable within the theory itself [5]. Rosser later strengthened the First Theorem by replacing Gödel's ω-consistency assumption with ordinary consistency [15]. Löb's theorem subsequently isolated the provability-theoretic mechanism underlying the Second Theorem [11].

These results are often compressed into the slogan that no sufficiently strong formal system can be both consistent and complete. The slogan is broadly correct, but it conceals distinctions that matter in a proof. A set of axioms may be computably enumerable without having a decidable axiom predicate. A primitive-recursive relation may be numeralwise represented in a weak theory even when that theory does not prove a uniform totality statement for an associated graph formula. A formula saying that one specified number is a proof is not the same as the existential formula saying that a sentence is provable. Ordinary consistency is weaker than 1-consistency, ω-consistency, and soundness.

The present article has three aims. First, it gives a proof-oriented route through syntax coding, proof verification, representability, diagonalization, Rosser's construction, and the derivability conditions. Second, it states the base-theory and effectiveness assumptions where they are used. Third, it separates the theorems from broader claims about human minds, artificial intelligence, or the impossibility of mathematical knowledge. Standard references provide fuller treatments of recursion theory and proof theory [1,7,10,17]; the purpose here is to display the central dependency chain in one ordinary mathematical paper, without claiming proof-assistant-level formalization.

2 Existing Expositions and the Present Contribution

Kleene and Hájek–Pudlák give technically authoritative developments, but their breadth can obscure the short chain of constructions needed by a first-time reader. Boolos, Burgess, and Jeffrey connect incompleteness with computability in a modern textbook framework; Smullyan emphasizes elegant fixed-point and provability arguments; Smith is especially careful pedagogically; and Franzén is indispensable for diagnosing philosophical misuse [1,3,7,10,17,18].

Short expositions commonly exhibit one or more of the following weaknesses: they treat a c.e. axiom set as though its membership predicate were primitive recursive; assert that proof checking is primitive recursive without displaying a bounded line checker; move from external computability to internal formalization without naming the arithmetic strength required; conflate numeralwise representation with uniform totality in $\mathbf{Q}$; conflate $\mathrm{Prf}_T(p, x)$ with $\mathrm{Prov}_T(x)$; state Gödel's original argument as though consistency alone established both unprovability and unrefutability; or state Tarski's theorem without specifying the standard model, coding, and truth biconditionals involved.

3 Notation and Logical Conventions

The standard natural numbers are denoted by $\mathbb{N}$. The numeral for $n \in \mathbb{N}$ is $\overline{n}$; the bar is omitted when no confusion is likely. We write $T \vdash \varphi$ for formal provability and $M \models \varphi$ for truth in a structure. The Gödel number of an expression φ is $\ulcorner \varphi \urcorner$; within arithmetic this notation abbreviates the corresponding numeral. A fixed contradiction, such as $0 = S0$, is denoted by $\bot$. The basic object language is

$$\mathcal{A} = \{0, S, +, \times, =\}.$$

Relative to these treatments, the distinctive feature of the present exposition is methodological rather than the introduction of a new incompleteness theorem. The present article makes the $\mathbf{Q}$-versus-$I\Sigma_1$ boundary a more persistent organizing convention than is customary in many textbook presentations; it does not claim that the cited texts state the relevant distinctions incorrectly. At each stage, the external metatheoretic assertion that a numerical operation or relation is primitive recursive is separated from the internal assertion that a specified arithmetic theory proves a representing formula. Fixed numerical computations and finite numeralwise verifications are assigned to $\mathbf{Q}$, whereas arguments requiring uniform quantification over arbitrary proof codes, witnesses, computation histories, or recursion lengths are carried out in $I\Sigma_1$. This boundary is not claimed to be proof-theoretically optimal. Its purpose is to prevent a metatheoretic family of finite verifications from being silently promoted to one internally proved universal statement.

Section 6 specifies the sequence code, exact proof calculus, capture-avoiding substitution, and proof verifier. Section 7 proves representability, including a finite-unfolding lemma and an explicit Σ_1 witness history. Section 8 distinguishes the external proof relation, its canonical internal formula, and the existential provability predicate. Section 12 gives detailed construction schemas for the uniform proof transformations needed for formalized Σ_1-completeness. Section 14 states Tarski's theorem as the nondefinability of the complete truth set of the standard model.

For bounded-formula arguments we work in the conservative definitional expansion by $\leq$ and $<$, where $x \leq y$ abbreviates $\exists z\,(z + x = y)$ and $x < y$ abbreviates $Sx \leq y$. Bounded quantifiers are written in the usual way. A Δ_0 formula belongs to the least syntactically defined class containing the atomic formulas and closed under Boolean connectives and bounded quantification. A Σ_1 formula is literally of the form $\exists u_1 \cdots \exists u_k\, \delta$, where $\delta \in \Delta_0$ and $k \geq 0$; Π_1 formulas are defined dually. These are primitive-recursively recognizable classes of formula codes. Saying that an arbitrary formula is merely equivalent to one in such a normal form is a separate metatheoretic assertion and is not used to define the induction scheme of $I\Sigma_1$.

A theory means a specified axiom presentation together with a fixed effective proof calculus, not merely its deductive closure. The metatheory is ordinary classical mathematics. Statements such as "f is primitive recursive" are external; statements such as $\mathbf{Q} \vdash \rho(\overline{a})$ concern formal derivability.

Symbol	Meaning
$\mathbb{N}$	standard natural numbers in the metatheory
$\overline{n}$	object-language numeral denoting $n \in \mathbb{N}$
$T \vdash \varphi$	φ is formally provable in T
$M \models \varphi$	φ is true in the structure M
$\ulcorner \varphi \urcorner$	Gödel number, or its numeral in the object language
$\mathrm{Prf}_T(p, x)$	p codes a T-proof of the formula coded by x
$\mathrm{Prov}_T(x)$	$\exists p\, \mathrm{Prf}_T(p, x)$
$\mathrm{Con}(T)$	$\neg\, \mathrm{Prov}_T(\ulcorner \bot \urcorner)$
$\beta(c, d, i)$	$c \bmod (1 + (i+1)d)$
$\mathrm{Th}(\mathbb{N})$	complete first-order truth set of the standard model

Table 1: Principal notation.

3.1 The fixed Hilbert calculus

To make the proof relation definite, fix the following classical Hilbert calculus for first-order logic with equality. Its primitive connectives are $\neg, \to, \forall$; the usual remaining connectives and $\exists$ are abbreviations. The logical axiom schemes are

(P1) $A \to (B \to A)$,

(P2) $(A \to (B \to C)) \to ((A \to B) \to (A \to C))$,

(P3) $(\neg B \to \neg A) \to (A \to B)$,

(Q1) $\forall x\, A \to A[t/x]$, when t is free for x in A,

(Q2) $\forall x\, (A \to B) \to (A \to \forall x\, B)$, when $x \notin \mathrm{FV}(A)$,

(E1) $t = t$,

(E2) $s = t \to (A[s/x] \to A[t/x])$,

where the displayed substitutions in (E2) are capture avoiding and free for the designated variable. The inference rules are modus ponens and generalization:

$$\frac{A \quad A \to B}{B}, \qquad \frac{A}{\forall x\, A}.$$

Because proofs have no undischarged assumptions, the generalization rule needs no additional side condition. A T-proof may also use the fixed axioms of $\mathbf{Q}$ and the designated nonlogical axioms of T. There are finitely many scheme types, and their instance predicates are checked by pattern matching, capture-avoiding substitution, and free-variable tests. This exact calculus will be used throughout; any standard recursively equivalent calculus would yield the same incompleteness results after routine recoding.

4 Scope and Exact Hypotheses

Definition 1 (Computably enumerable axiomatization)**:** *A theory T is* computably enumerable *(c.e.), or* effectively axiomatized*, if an algorithm enumerates the Gödel numbers of its nonlogical axioms.*

Definition 2 (Primitive-recursive axiomatization)**:** *A theory T is* primitive-recursively axiomatized *if membership in its nonlogical axiom set is decided by a primitive-recursive predicate* $\mathrm{Ax}_T(x)$.

Every primitive-recursively axiomatized theory is c.e., but the converse fails. For a merely c.e. theory, one uses a primitive-recursive stage predicate $\mathrm{Ax}_T(a, s)$ saying that a fixed deterministic enumerator has output a during its first s steps. A proof record carries a stage certificate for every additional nonlogical axiom line. Section 6.3 proves that the enriched verifier is primitive recursive and recognizes exactly the ordinary finite T-proofs.

Definition 3 (Consistency and correctness conditions)**:** *Let T be a classical theory.*

(i) T *is* consistent *if* $T \nvdash \bot$.
(ii) T *is* ω-consistent *if there is no formula* $\varphi(x)$ *such that* $T \vdash \exists x\, \varphi(x)$ *and* $T \vdash \neg\varphi(\overline{n})$ *for every* $n \in \mathbb{N}$.
(iii) T *is* 1-consistent *if it proves no false* Σ_1 *sentence.*
(iv) T *is* sound *if every theorem of* T *is true in* $\mathbb{N}$.

For arithmetical theories extending $\mathbf{Q}$,

$$\text{soundness} \implies \omega\text{-consistency} \implies \text{1-consistency} \implies \text{consistency}.$$

Indeed, suppose an ω-consistent extension of $\mathbf{Q}$ proved a false Σ_1 sentence $\exists u_1 \cdots \exists u_k\, \delta(u_1, \dots, u_k)$ with $\delta \in \Delta_0$. After primitive-recursive coding of the witness tuple this is equivalent, provably in $\mathbf{Q}$, to a one-quantifier sentence $\exists x\, \delta'(x)$ with $\delta' \in \Delta_0$, where $\delta'(x)$ asserts that x codes a tuple satisfying δ. Since the sentence is false, every standard instance $\delta'(\overline{n})$ is false and $\mathbf{Q}$ proves its negation, contradicting ω-consistency. The same tuple-coding step is used later in Lemma 32.

Theorem 4 (First Incompleteness Theorem, Rosser form)**:** *Let T be a consistent c.e. theory in the language of arithmetic with $T \supseteq \mathbf{Q}$. Then there is an arithmetical sentence R_T such that*

$$T \nvdash R_T \quad \textit{and} \quad T \nvdash \neg R_T.$$

Remark 5 (Interpretation version)**:** *A more general formulation allows a theory in another language that effectively interprets an adequate arithmetic theory. The undecidable sentence is then the translation of an arithmetical sentence through the interpretation. To avoid hiding that translation, the proofs below use the direct hypothesis $T \supseteq \mathbf{Q}$.*

Theorem 6 (Second Incompleteness Theorem, convenient standard form)**:** *Let T be a consistent c.e. extension of $I\Sigma_1$, equipped with the canonical proof predicate constructed below. With*

$$\mathrm{Con}(T) \equiv \neg\,\mathrm{Prov}_T(\ulcorner \bot \urcorner),$$

one has $T \nvdash \mathrm{Con}(T)$.

The assumption $T \supseteq I\Sigma_1$ is sufficient rather than sharp. Abstractly, what is needed is a provability predicate satisfying the Hilbert–Bernays–Löb derivability conditions. The use of $I\Sigma_1$ makes the required uniform proof transformations transparent.

5 Robinson Arithmetic and Peano Arithmetic

Robinson arithmetic $\mathbf{Q}$ has the finite axioms

$$\begin{aligned} &Sx \neq 0, \qquad Sx = Sy \to x = y, \qquad x \neq 0 \to \exists y\, x = Sy,\\ &x + 0 = x, \qquad x + Sy = S(x + y),\\ &x \times 0 = 0, \qquad x \times Sy = x \times y + x. \end{aligned}$$

It has no induction schema. Nevertheless, it proves every true closed Δ_0 sentence and can verify every fixed correct primitive-recursive computation numeral by numeral. This decidability of closed Δ_0 sentences is used repeatedly below, so we record it as a lemma.

Lemma 7 (Closed Δ_0-completeness of $\mathbf{Q}$)**:** *For every closed Δ_0 sentence δ, if $\mathbb{N} \models \delta$ then $\mathbf{Q} \vdash \delta$, and if $\mathbb{N} \models \neg\delta$ then $\mathbf{Q} \vdash \neg\delta$.*

Proof: The argument is an external structural induction in the metatheory; it is not an internal induction in $\mathbf{Q}$. First, every closed term t evaluates to a unique numeral: by external induction on term structure, using $x + 0 = x$, $x + Sy = S(x + y)$, $x \times 0 = 0$, and $x \times Sy = x \times y + x$, there is $n \in \mathbb{N}$ with $\mathbf{Q} \vdash t = \overline{n}$. For each standard m, $\mathbf{Q}$ also proves the numeral classification

$$z < \overline{m} \leftrightarrow (z = \overline{0} \vee \cdots \vee z = \overline{m-1}),$$

by external induction on m from the predecessor axiom $x \neq 0 \to \exists y\,(x = Sy)$, successor injectivity, and the definition of $<$; the case $m = 0$ uses $Sx \neq 0$.

Now induct externally on the construction of the closed Δ_0 sentence δ. For an atomic sentence $s = t$, let $\overline{a}, \overline{b}$ be the numeral values of s, t. If $\mathbb{N} \models s = t$ then $a = b$, so $\mathbf{Q} \vdash s = \overline{a}$ and $\mathbf{Q} \vdash t = \overline{a}$ give $\mathbf{Q} \vdash s = t$. If $\mathbb{N} \models s \neq t$ then $a \neq b$, and $\mathbf{Q} \vdash \overline{a} \neq \overline{b}$ follows by external induction using $Sx \neq 0$ and successor injectivity, whence $\mathbf{Q} \vdash s \neq t$. Boolean combinations follow from the induction hypothesis by propositional logic. For a bounded sentence $\exists z < \overline{m}\,\eta(z)$, the classification reduces it, provably in $\mathbf{Q}$, to $\bigvee_{i<m} \eta(\overline{i})$; if it is true some disjunct is true and is proved by the induction hypothesis, and if it is false every disjunct is false and is refuted by the induction hypothesis. The bounded universal case is dual. □

Peano Arithmetic **PA** adds induction

$$(\varphi(0) \wedge \forall x\,(\varphi(x) \to \varphi(Sx))) \to \forall x\,\varphi(x)$$

for every formula φ. The fragment $I\Sigma_1$ restricts induction to Σ_1 formulas. It is weaker than **PA**, but strong enough to prove totality and correctness of the primitive-recursive proof-code transformations used later.

Methodological rule.

Use **Q** for numeralwise verification of fixed computations. Use $I\Sigma_1$ when a proof requires a uniform statement about all proof codes, all witnesses, or all recursion lengths.

6 Arithmetization of Syntax

6.1 A concrete Gödel-style prime-power sequence code

Following Gödel's original construction, code a finite sequence $a_0, \dots, a_{k-1}$ by

$$[a_0, \dots, a_{k-1}] := \prod_{i<k} p_i^{a_i+1},$$

where p_i is the i-th prime and the empty sequence has code 1. The exponent shift prevents a final zero entry from disappearing. Unique factorization makes the coding injective.

Let $v(i, s)$ be the largest $e \le s$ such that p_i^e divides s, with value 0 when p_i does not divide s. The prime function, exponentiation, divisibility, and bounded maximization are primitive recursive, hence so is v. Put $\mathrm{lh}(0) = 0$, and for $s > 0$ define

$$\mathrm{lh}(s) = \mu\, i \le s\,[\, p_i \nmid s\,].$$

The notation $\mu\, i \le s\,[P(i)]$ means the least i in $\{0, \dots, s\}$ satisfying $P(i)$. This is bounded minimization: one tests $0, 1, \dots, s$ in order, so the operation is primitive recursive. In this application the set is nonempty, since $p_s > s$ and therefore $p_s \nmid s$ whenever $s > 0$. Thus the bounded minimum is defined for every $s > 0$. Set

$$(s)_i = \begin{cases} v(i, s) - 1, & i < \mathrm{lh}(s), \\ 0, & i \ge \mathrm{lh}(s). \end{cases}$$

Finally, let $\mathrm{Seq}(s)$ assert that $s > 0$ and

$$s = \prod_{i<\mathrm{lh}(s)} p_i^{(s)_i+1}.$$

The bounded product is primitive recursive. Thus Seq, lh, and entry extraction are primitive recursive. On valid codes,

$$\mathrm{append}(s, a) = s \cdot p_{\mathrm{lh}(s)}^{a+1}$$

is primitive recursive. A pair is the two-entry sequence

$\langle a, b\rangle = [a, b] = 2^{a+1}3^{b+1}$;

its projections are shifted exponents of 2 and 3. Every component of a valid nonempty sequence code is smaller than the code.

Remark 8 (Interchangeability of codings)**:** *Prime-power coding is historically recognizable and makes unique decodability transparent. Polynomial pairing, cons/nil lists, or β-coded lists yield the same metamathematical results after routine changes to numerical definitions. Section 7.6 uses a polynomial pair only inside one bounded formula, because its graph is quantifier free in $\mathcal{A}$.*

Lemma 9 (Concrete sequence and syntax coding)**:** *The predicates and functions* $\mathrm{Seq}(q)$, $\mathrm{lh}(q)$, *and* $(q)_i$ *are primitive recursive. There is a concrete coding of variables, terms, and formulas for which the following are primitive recursive:*

(i) $\mathrm{Term}(e)$, $\mathrm{Fm}(e)$, *and* $\mathrm{Sent}(e)$*;*
(ii) constructor functions for $0, S, +, \times, =, \neg, \rightarrow, \forall$*, together with inverse pattern tests;*
(iii) occurrence, free-variable, and maximum-variable-index functions;
(iv) the instance predicates for the finitely many logical and equality axiom schemes fixed in Section 6.

Proof: Reserve tags

tag	0	1	2	3	4	5	6	7	8	9
constructor	Var	0	S	$+$	$\times$	$=$	$\neg$	$\rightarrow$	$\forall$	Aux

and code a node with tag t and children $e_0, \dots, e_{r-1}$ by

$E(t; e_0, \dots, e_{r-1}) = 1 + \langle t, [e_0, \dots, e_{r-1}]\rangle$.

Thus $x_i = E(0; i)$, the constant $0 = E(1;)$, $S\tau = E(2; \tau)$, and so forth. The outer pair and child-sequence code strictly dominate their components, so every immediate subexpression has smaller code than the whole expression. A course-of-values recursion on e decides formation by inspecting the outer tag, arity, and previously computed formation values of children. Course-of-values recursion reduces to ordinary primitive recursion by storing the preceding finite table in a prime-power sequence.

The same recursion computes variable occurrence, free variables, and maximum variable index. At $\forall x_i \varphi$, the occurrence set adds i, while the free-variable set removes i from $\mathrm{FV}(\varphi)$. Finite-set operations are primitive recursive under the sequence coding. Each logical or equality axiom instance is recognized by a finite tag test, pattern matching, and the substitution and free-variable tests established in Lemma 10. A finite disjunction over the fixed scheme types remains primitive recursive. □

6.2 Capture-avoiding substitution with an explicit majorant

The coding uses named variables $x_0, x_1, \ldots$. We first define a numerical operation $\mathrm{RenF}(e, j, r)$ by structural recursion: it changes free occurrences of x_j in e to x_r, leaves other variables and 0 fixed, commutes with term and propositional constructors, and satisfies

$$\mathrm{RenF}(\forall x_k \varphi, j, r) = \begin{cases} \forall x_k \varphi, & k = j, \\ \forall x_k \, \mathrm{RenF}(\varphi, j, r), & k \neq j. \end{cases}$$

As a total numerical function this recursion is defined for all inputs. Its intended alpha-renaming property is asserted only under the side condition $r \notin \mathrm{Var}(e)$; without that condition, a newly introduced x_r could be captured by an intervening binder.

Define

$$\mathrm{fresh}(\varphi, t, v, j) = 1 + \max\{v, j, \mathrm{maxvar}(\varphi), \mathrm{maxvar}(t)\}.$$

The substitution $\mathrm{Sub}(e, v, t)$ replaces free occurrences of x_v by the term t, leaves other variables and 0 fixed, and commutes with term constructors, atomic formulas, and propositional connectives. At a quantifier,

$$\mathrm{Sub}(\forall x_j \varphi, v, t) = \begin{cases} \forall x_j \varphi, & j = v, \\ \forall x_j \, \mathrm{Sub}(\varphi, v, t), & j \neq v \text{ and } j \notin \mathrm{FV}(t), \\ \forall x_r \, \mathrm{Sub}(\mathrm{RenF}(\varphi, j, r), v, t), & j \neq v \text{ and } j \in \mathrm{FV}(t), \end{cases}$$

where $r = \mathrm{fresh}(\varphi, t, v, j)$ in the last case. This r occurs in neither φ nor t, so the call to RenF satisfies its freshness side condition.

The numeral-code function is primitive recursive:

$$\mathrm{num}(0) = \ulcorner 0 \urcorner, \qquad \mathrm{num}(n+1) = E(2; \mathrm{num}(n)).$$

Consequently, substitution of numerals, formation of negations, and diagonal substitution are primitive recursive.

Lemma 10 (Primitive-recursive capture-avoiding substitution)**:** *The numerical functions* RenF, fresh, *and* Sub *are primitive recursive. If e codes a term or formula and t codes a term, then* $\mathrm{Sub}(e, v, t)$ *codes capture-avoiding substitution of t for the free occurrences of x_v in e. If $r \notin \mathrm{Var}(e)$, then* $\mathrm{RenF}(e, j, r)$ *has the intended fresh-renaming semantics.*

Proof: The semantic assertions follow by structural induction. The stopping clause at a binder $\forall x_j$ prevents alteration of occurrences bound by that binder. In the capture-risk case, the chosen index

$$r = 1 + \max\{v, j, \mathrm{maxvar}(\varphi), \mathrm{maxvar}(t)\}$$

does not occur in the body or the inserted term, so alpha-renaming the binder to x_r is legitimate and no free variable of t becomes captured.

For primitive recursiveness, let $\mathrm{nd}(e)$ be the node count and put, for $z \geq 10$,

$$H(z) = 1 + 2^{z+1}3^{2^{z+1}3^{z+1}+1}.$$

If every tag, variable index, and immediate child code is at most z, then every constructor node of arity at most two has code at most $H(z)$. Write $H^{[m]}$ for the m-fold iterate. Define the two separate bounds

$$B_{\mathrm{RenF}}(e, j, r) = 1 + \max\{e, j, r, 10\}, \qquad M_{\mathrm{RenF}}(e, j, r) = H^{[\mathrm{nd}(e)+1]}(B_{\mathrm{RenF}}(e, j, r)),$$
$$B_{\mathrm{Sub}}(e, v, t) = 1 + \max\{e, v, t, 10\}, \qquad M_{\mathrm{Sub}}(e, v, t) = H^{[\mathrm{nd}(e)(\mathrm{nd}(t)+1)+2]}(B_{\mathrm{Sub}}(e, v, t)).$$

All four functions are primitive recursive. On well-formed inputs, every variable index already occurring in e or t is below the corresponding expression code; in the capture-risk clause the fresh index satisfies $r \leq B_{\mathrm{Sub}}(e, v, t)$. Renaming preserves the tree shape, so M_{RenF} bounds every intermediate and final renaming code. A substitution output has at most $\mathrm{nd}(e)(\mathrm{nd}(t) + 1)$ nodes: each variable node of e contributes at most one copy of t, while fresh renaming adds no node. Hence M_{Sub} bounds every intermediate and final substitution code.

Implement both operations with a fuel parameter bounded by $\mathrm{nd}(e)$, returning a fixed default on malformed input or exhausted fuel. Every recursive call decreases the fuel; tag tests, free-variable membership, fresh-index computation, and constructor formation are primitive recursive. Bounded recursion using the displayed majorants is therefore primitive recursive. Substituting the primitive-recursive fuel value yields the required total numerical functions. □

6.3 Why the proof relation is primitive recursive

A proof is a coded sequence of line records

$$\langle a, \langle t, \langle u, \langle v, s\rangle\rangle\rangle\rangle,$$

where a is a formula code, t is one of the six justification tags

LogAx, QAx, IndAx, ExtraAx, MP, Gen,

u, v are possible indices of earlier lines, and s is a possible stage certificate. The tag IndAx marks an instance of the Σ_1-induction scheme and is used only for theories containing $I\Sigma_1$; for theories without induction it never occurs. Unused fields are 0. Record projections and the predicate $\mathrm{SeqProof}(p)$ saying that p is a sequence of correctly shaped records are primitive recursive.

For a fixed deterministic enumerator E_T, code machine configurations by natural numbers so that the one-step transition $\mathrm{step}_T(c)$ is primitive recursive. A configuration need not emit an axiom at every step,

so emission is recorded by a primitive-recursive relation $\mathrm{Out}_T(c, a)$ holding exactly when configuration c emits the formula coded by a. Define by primitive recursion

$$\mathrm{conf}_T(0) = c_0, \qquad \mathrm{conf}_T(s+1) = \mathrm{step}_T(\mathrm{conf}_T(s)).$$

Then

$$\mathrm{Ax}_T(a, s) \iff \exists i \le s\, \mathrm{Out}_T(\mathrm{conf}_T(i), a)$$

is primitive recursive by bounded search. Using the relation Out_T rather than an output function avoids assigning a spurious axiom code to a non-emitting step; equivalently one may fix a distinguished no-output value such as 0, which under the coding of Section 6 is not a formula code. This direct simulation avoids any unspoken bound on codes of computation histories.

Theorem 11 (Primitive recursiveness of proof verification)**:** *Let T use the fixed calculus of Section 6.*

(i) If its additional axiom predicate $\mathrm{Ax}_T(a)$ is primitive recursive, then the relation $\mathrm{Prf}_T^{\mathrm{ext}}(p, x)$, meaning that p is a valid T-proof ending with the formula coded by x, is primitive recursive.

(ii) If the additional axioms are merely c.e., the stage predicate above and witness-annotated proof records yield a primitive-recursive relation $\mathrm{Prf}_T^{\mathrm{ext}}(p, x)$ recognizing exactly the ordinary finite T-proofs.

Proof: Let $\mathrm{fm}(r)$ be the formula field of a record and $a_i = \mathrm{fm}((p)_i)$. Define $\mathrm{LineOK}_T(p, i)$ as the finite disjunction of the following exact cases:

(a) a_i is an instance of one of (P1)–(E2);
(b) a_i is one of the seven fixed axioms of **Q**;
(c) a_i is an additional T-axiom, verified either by $\mathrm{Ax}_T(a_i)$ or, in the c.e. case, by the stored stage certificate s_i via $\mathrm{Ax}_T(a_i, s_i)$;
(d) a_i carries the tag IndAx and is an instance of the Σ_1-induction scheme, recognized by the primitive-recursive syntactic test for that fixed scheme; this clause is present only when $T \supseteq I\Sigma_1$ and is vacuous otherwise;
(e) the record stores $j, k < i$, line j has code A, line k has code $A \to B$, and a_i is the code of B;
(f) the record stores $j < i$, a variable index r, and a_i is the constructor code of $\forall x_r A$, where line j has code A.

Every clause is primitive recursive by Lemmas 9 and 10; all searches over predecessor lines are bounded by i. Put

$$\mathrm{Prf}_T^{\mathrm{ext}}(p, x) \iff \mathrm{SeqProof}(p) \wedge \mathrm{lh}(p) > 0 \wedge \mathrm{Fm}(x) \wedge \mathrm{fm}((p)_{\mathrm{lh}(p)-1}) = x \wedge \forall i < \mathrm{lh}(p)\, \mathrm{LineOK}_T(p, i).$$

The final universal quantifier is bounded, so the relation is primitive recursive.

In the c.e. case, the two recognition directions are tight. For soundness of the verifier, clause (c) accepts an additional-axiom line only together with a stored certificate s_i satisfying $\mathrm{Ax}_T(a_i, s_i)$, i.e. only when the enumerator has actually output a_i by stage s_i; hence every accepted line cites a genuine enumerated

axiom. For completeness, every ordinary finite proof has stage certificates because each of its finitely many additional axioms eventually appears in the fixed enumeration, so a sufficiently large stage witnesses it. Erasing certificates maps every enriched proof to an ordinary proof, and every ordinary proof has at least one enriched version. Thus the two notions recognize exactly the same theorems, but the erasure map is generally many-to-one. Choosing the least valid stage certificate on each extra-axiom line gives a canonical section of erasure, not a two-sided inverse. □

Remark 12: *Theorem 11 is an external computability theorem. It does not yet choose an object-language formula with the uniform nonstandard behavior required by the derivability conditions. Sections 7 and 8 address those two distinct tasks.*

7 Primitive-Recursive Functions and Representability

7.1 Primitive recursion and numeralwise representation

Primitive-recursive functions are generated from zero, successor, and projections by composition and primitive recursion. A relation is primitive recursive when its characteristic function is primitive recursive. The class is closed under Boolean operations and bounded quantification.

Definition 13 (Numeralwise representation)**:** *A formula $\rho(\vec{x})$* numeralwise represents *$R \subseteq \mathbb{N}^k$ in T if, for every $\vec{a} \in \mathbb{N}^k$,*

$$R(\vec{a}) \Rightarrow T \vdash \rho(\overline{\vec{a}}), \qquad \neg R(\vec{a}) \Rightarrow T \vdash \neg\rho(\overline{\vec{a}}).$$

Definition 14 (Strong numeralwise representation)**:** *A formula $F(\vec{x}, y)$* strongly numeralwise represents *a total function $f : \mathbb{N}^k \to \mathbb{N}$ in T if, whenever $f(\vec{a}) = b$,*

$$T \vdash \forall y\,(F(\overline{\vec{a}}, y) \leftrightarrow y = \overline{b}).$$

This is a family of theorems indexed by standard inputs, not the uniform assertion $T \vdash \forall \vec{x}\,\exists! y\,F(\vec{x}, y)$.

Theorem 15 (Representability theorem)**:** *Every primitive-recursive function is strongly numeralwise representable in* **Q**. *Every primitive-recursive relation is numeralwise representable in* **Q**.

7.2 Initial functions and composition

The zero, successor, and projection functions are represented by

$$y = 0, \qquad y = Sx, \qquad y = x_i.$$

For each fixed input tuple, **Q** proves that these formulas have exactly the intended numeral as output. Suppose $g_1, \dots, g_m$ are strongly represented by $G_1, \dots, G_m$, and h by H. For

$$f(\vec{x}) = h(g_1(\vec{x}), \dots, g_m(\vec{x})),$$

define

$$F(\vec{x}, y) \equiv \exists z_1 \cdots \exists z_m \left(\bigwedge_{i=1}^{m} G_i(\vec{x}, z_i) \wedge H(z_1, \dots, z_m, y) \right).$$

Fix standard $\vec{a}$, let $b_i = g_i(\vec{a})$, and $c = h(b_1, \dots, b_m)$. The strong representation of each G_i reduces z_i to $\overline{b_i}$ inside $\mathbf{Q}$, after which the strong representation of H yields

$$\mathbf{Q} \vdash \forall y \, (F(\overline{\vec{a}}, y) \leftrightarrow y = \overline{c}).$$

7.3 The β-function and an internally functional entry formula

Define externally

$$\beta(c, d, i) = c \bmod (1 + (i + 1)d).$$

Lemma 16 (β-function lemma)**:** *For every finite sequence $a_0, \dots, a_n$ there are $c, d \in \mathbb{N}$ such that $\beta(c, d, i) = a_i$ for $0 \le i \le n$.*

Proof: Choose $s \ge \max\{n, a_0, \dots, a_n, 2\}$, set $d = s!$, and put $m_i = 1 + (i + 1)d$. Suppose a prime p divided both m_i and m_j, with $i < j \le n$. Then $p \mid (j - i)d$. Since $p \mid m_i = 1 + (i + 1)d$, one has $p \nmid d$; hence $p \mid j - i$. But $1 \le j - i \le n \le s$, so $p \mid s! = d$, a contradiction. Thus the m_i are pairwise coprime. Moreover, $a_i < m_i$. The Chinese Remainder Theorem supplies c satisfying $c \equiv a_i \pmod{m_i}$, and the least remainder is a_i. □

Let $M(d, i) = 1 + (i + 1)d$ and define

$$B(c, d, i, w) \equiv \exists q \le c \, [c = qM(d, i) + w \wedge w < M(d, i)].$$

This is Δ_0. Define the uniqueness-augmented formula

$$B^!(c, d, i, w) \equiv B(c, d, i, w) \wedge \forall r < M(d, i) \, [B(c, d, i, r) \to r = w].$$

Lemma 17 (Numeralwise correctness and internal functionality of $B^!$)**:** *(i) If $\beta(c, d, i) = w$ for standard c, d, i, w, then $\mathbf{Q} \vdash B^!(\overline{c}, \overline{d}, \overline{i}, \overline{w})$.*
(ii) $\mathbf{Q} \vdash B^!(c, d, i, u) \wedge B^!(c, d, i, v) \to u = v$.

Proof: For (i), every parameter and bound is a numeral, so $B^!(\overline{c}, \overline{d}, \overline{i}, \overline{w})$ is a closed Δ_0 sentence. The quotient-and-remainder equation and the finitely many competing remainder cases are true closed Δ_0 facts, so by Lemma 7 $\mathbf{Q}$ proves the correct one and refutes the others. For (ii), the uniqueness conjunct of the first instance applies to the remainder supplied by the second; the required bound on v is part of $B(c, d, i, v)$. □

7.4 Finite unfolding and fixed-numeral comparison in Q

Lemma 18 (Finite bounded unfolding and comparison)**:** *For every standard m and every formula $\theta(i, \vec{z})$,* Q *proves, uniformly in the remaining free variables,*

$$\forall i < \overline{m}\, \theta(i, \vec{z}) \leftrightarrow \bigwedge_{r<m} \theta(\overline{r}, \vec{z}), \qquad \exists i < \overline{m}\, \theta(i, \vec{z}) \leftrightarrow \bigvee_{r<m} \theta(\overline{r}, \vec{z}).$$

The empty conjunction is $\top$ and the empty disjunction is $\bot$. With the order conventions of Section 6, Q *also proves, for every standard m,*

$$x < \overline{m} \leftrightarrow (x = \overline{0} \vee \cdots \vee x = \overline{m-1}), \qquad x < \overline{m} \vee \overline{m} \leq x. \tag{1}$$

Proof: An external induction on m constructs a separate finite **Q**-derivation for each numeral bound. The case $m = 0$ uses the axiom that no successor is 0. In the successor step, the predecessor axiom $x \neq 0 \rightarrow \exists y\,(x = Sy)$, together with injectivity of successor, splits an arbitrary x into the new numeral case and the previously treated cases. Iterating this finite argument yields the displayed classification of $x < \overline{m}$. The same m-step case analysis gives either $x < \overline{m}$ or $x = S^m y$ for some y. For fixed m, finitely many applications of the addition equations prove $\mathbf{Q} \vdash y + \overline{m} = S^m y$; hence the latter case implies $\overline{m} \leq x$ under the definition $\exists z\,(z + \overline{m} = x)$. Substituting the finite classification into a bounded quantifier and using ordinary first-order logic gives the two unfolding equivalences. No induction schema is used inside **Q**. □

7.5 Primitive recursion and completion of Theorem 15

Suppose

$$f(0, \vec{x}) = g(\vec{x}), \qquad f(n+1, \vec{x}) = h(n, f(n, \vec{x}), \vec{x}),$$

and G, H strongly numeralwise represent g, h. Define

$$\begin{aligned} F(n, \vec{x}, y) \equiv \exists c\, \exists d\, \Big[& \exists u\,(B^!(c, d, 0, u) \wedge G(\vec{x}, u)) \\ & \wedge \forall i < n\, \exists u\, \exists v\,(B^!(c, d, i, u) \wedge B^!(c, d, i+1, v) \wedge H(i, u, \vec{x}, v)) \\ & \wedge B^!(c, d, n, y) \Big]. \end{aligned}$$

Fix standard $m, \vec{a}$, compute $b_i = f(i, \vec{a})$ for $i \leq m$, and choose standard c, d coding these values by Lemma 16. Lemmas 17 and 18, together with the representation hypotheses for G, H, give $\mathbf{Q} \vdash F(\overline{m}, \overline{\vec{a}}, \overline{b_m})$. Conversely, assuming $F(\overline{m}, \overline{\vec{a}}, y)$, unfold the bounded universal. Strong representation fixes the initial value; internal functionality of $B^!$ identifies duplicate entries at a given index; and the representation of H fixes each successor value. Repeating this finite argument through the m transitions gives

$$\mathbf{Q} \vdash \forall y\,(F(\overline{m}, \overline{\vec{a}}, y) \leftrightarrow y = \overline{b_m}).$$

This is a different finite proof for each standard m, not an induction over arbitrary computation lengths inside **Q**.

The initial functions, composition, and primitive recursion are therefore closed under strong numeralwise representation. For a primitive-recursive relation R, let $\chi_R(\vec{a}) = 0$ exactly when $R(\vec{a})$, strongly represent χ_R by C_R, and put $\rho_R(\vec{x}) \equiv C_R(\vec{x}, 0)$. This completes the proof of Theorem 15.

7.6 Explicit Σ_1 closure and a witness-history formula

The initial graph formulas are quantifier free. Closure under composition preserves Σ_1 form explicitly: if

$$G_i(\vec{x}, z_i) \equiv \exists u_i\, \delta_i(\vec{x}, z_i, u_i), \qquad H(\vec{z}, y) \equiv \exists v\, \eta(\vec{z}, y, v),$$

with $\delta_i, \eta \in \Delta_0$, then the graph of the composition is

$$\exists \vec{z}\, \exists \vec{u}\, \exists v \left(\bigwedge_i \delta_i(\vec{x}, z_i, u_i) \wedge \eta(\vec{z}, y, v) \right),$$

again Σ_1.

For primitive recursion, the preceding formula F contains a bounded universal whose matrix can contain the Σ_1 formula H. All witnesses for the bounded family must therefore be packed into one finite code. Assume, after tuple coding, that

$$G(\vec{x}, u) \equiv \exists a\, \delta_G(\vec{x}, u, a), \qquad H(i, u, \vec{x}, v) \equiv \exists b\, \delta_H(i, u, \vec{x}, v, b),$$

with bounded matrices. For witness records use the polynomial pairing relation

$$\mathrm{Pair}(a, b, r) \equiv r = (a + b)(a + b) + a.$$

It is quantifier free and injective: pairs with fixed sum $a + b = t$ occupy the interval $[t^2, t^2 + t]$, and these intervals are disjoint.

Let e, f be β-parameters storing witness records. Define

$$\begin{aligned}
\mathrm{Hist}_{G,H}(c, d, e, f, K, n, \vec{x}, y) \equiv \exists u \le K\, \exists a \le K\, \exists r \le K\, [&B^!(e, f, 0, r) \\
&\wedge \mathrm{Pair}(u, a, r) \wedge B^!(c, d, 0, u) \wedge \delta_G(\vec{x}, u, a)] \\
\wedge\, \forall i < n\, \exists u, v, b, q, r \le K\, [&B^!(e, f, i + 1, r) \\
&\wedge \mathrm{Pair}(v, b, q) \wedge \mathrm{Pair}(u, q, r) \wedge B^!(c, d, i, u) \\
&\wedge B^!(c, d, i + 1, v) \wedge \delta_H(i, u, \vec{x}, v, b)] \\
\wedge\, &B^!(c, d, n, y).
\end{aligned}$$

Here $\exists u, v, b, q, r \le K$ abbreviates five bounded quantifiers. Every quantifier in $\mathrm{Hist}_{G,H}$ is bounded. Put

$$F_\Sigma(n, \vec{x}, y) \equiv \exists c\, \exists d\, \exists e\, \exists f\, \exists K\, \mathrm{Hist}_{G,H}(c, d, e, f, K, n, \vec{x}, y). \tag{2}$$

Lemma 19 (Σ_1 form of representability)**:** *The formula in* (2) *is* Σ_1 *and strongly numeralwise represents the function defined by primitive recursion from* g, h*. Consequently, every primitive-recursive function has a* Σ_1 *graph representer in* $\mathbf{Q}$*, and every primitive-recursive relation has a* Σ_1 *numeralwise representer.*

Proof: The matrix is bounded, so (2) is Σ_1. For fixed standard inputs, choose c, d coding the actual computation values, witnesses for the initial and transition graph instances, a bound K larger than all values and record codes, and e, f coding the finite record sequence by Lemma 16. Lemmas 17 and 18 give the positive numeral instance in $\mathbf{Q}$. For uniqueness, the witness record is irrelevant once existence is established: the c, d entries are internally functional, and the strong representation of G, H fixes successive values by the same finite argument used above. Together with the explicit composition closure and quantifier-free initial graphs, this completes the structural induction for Σ_1 graph formulas. □

8 Proof and Provability Predicates

Fix a c.e. theory $T \supseteq \mathbf{Q}$. Let $\mathrm{Prf}_T^{\mathrm{ext}}(p, x)$ be the external primitive-recursive relation from Theorem 11. Lemma 19 supplies Σ_1 formulas that numeralwise represent it. For the First Theorem, that much is enough. The derivability conditions require a particular formula whose behavior on nonstandard inputs reflects the actual proof checker and its proof transformations.

Definition 20 (Canonical normalized proof predicate)**:** *Normalize the axiom presentation so that logical axioms, the seven axioms of* $\mathbf{Q}$*, the induction axioms of* $I\Sigma_1$ *when applicable, and additional* T*-axioms have separate line tags, in agreement with the* IndAx *tag of Section 6.3. Here the induction scheme ranges over the syntactic* Σ_1 *class defined in Section 3 and recognized as a primitive-recursive class of formula codes in Section 6. Additional merely c.e. axioms retain stage certificates. The formula* $\mathrm{Prf}_T(p, x)$ *is the canonical* Σ_1 *trace formula obtained by arithmetizing this exact witness-annotated checker. Every auxiliary graph formula used for sequence transformations, proof concatenation, retagging, numeral substitution, and proof synthesis is obtained from the same trace convention.*

Lemma 21 (Canonical trace correctness; detailed construction schema)**:** *For every primitive-recursive constructor or proof transformation used below, there is a canonical* Σ_1 *graph formula* $\Gamma_f(\vec{x}, y) \equiv \exists h\, \mathrm{Trace}_f(\vec{x}, y, h)$ *with bounded trace matrix, such that:*

(i) $\mathbf{Q}$ *proves the correct positive or negative instance for every fixed standard tuple;*

(ii) $I\Sigma_1$ *proves* $\forall \vec{x}\, \exists! y\, \Gamma_f(\vec{x}, y)$ *and the defining equations of* f*;*

(iii) $I\Sigma_1$ *proves the local correctness implications needed for sequence concatenation, line retagging, proof lifting, modus-ponens extension, and existential-introduction extension.*

In particular, $I\Sigma_1$ proves the internal correctness statements invoked in Section 12. The claim is at ordinary paper-level granularity: the recursive trace clauses, majorants, and induction invariant are specified, while the complete Gödel-number formula for every local constructor is not printed.

Proof: Compile a primitive-recursive definition into a finite instruction tree whose nodes are initial functions, composition, and primitive recursion. A trace records the values and subsidiary traces at each instruction. For composition, the trace contains the intermediate outputs. For primitive recursion, it contains a finite value history together with traces for the initial and step computations. Every node of the instruction tree contributes a bounded block to the trace. A primitive-recursive majorant for the recorded values and for the whole trace code is defined simultaneously by structural recursion on the chosen primitive-recursive definition of f, mirroring the value-bounding recursions used in Section 7: at an initial node it is the obvious value bound; at a composition node it is obtained from the majorants of the components; and at a primitive-recursion node it iterates the step majorant along the recursion length. Hence the entire trace code is bounded by a primitive-recursive function of $\vec{x}$ and y. All local trace checks are bounded once the trace code and output are supplied, so Trace_f is Δ_0 and the single existential quantifier over h keeps Γ_f at the Σ_1 level rather than merely arithmetical. Numeralwise correctness follows from the representability construction of Section 7.

For internal totality and functionality, $I\Sigma_1$ proves the initial and composition clauses directly and proves the primitive-recursion clause by Σ_1-induction on the recursion length, with the partial trace as existential witness. Proof-sequence transformations are particular primitive recursions on a coded length. Their local correctness is bounded: each copied line preserves its tag and predecessors, each retagged base axiom is accepted by the target checker, and each appended inference satisfies the corresponding line clause. A Σ_1-induction on the processed prefix proves the stated uniform implications. □

For every standard p, x,

$$\mathrm{Prf}_T^{\mathrm{ext}}(p,x) \iff \mathbb{N} \vDash \mathrm{Prf}_T(\overline{p},\overline{x}),$$

and $\mathbf{Q}$ proves the appropriate positive or negative numeral instance. Define

$$\mathrm{Prov}_T(x) \equiv \exists p\, \mathrm{Prf}_T(p,x).$$

Because Prf_T is Σ_1, so is Prov_T.

The distinction is fundamental. $\mathrm{Prf}_T(p, \ulcorner A \urcorner)$ says that one specified number is a proof of A; every standard instance is decidable in $\mathbf{Q}$. The formula $\mathrm{Prov}_T(\ulcorner A \urcorner)$ says that some proof code exists. When no proof exists, there is no general reason for T to prove $\forall p\, \neg \mathrm{Prf}_T(p, \ulcorner A \urcorner)$.

Proposition 22 (Positive representability of provability)**:** *If $T \vdash A$, then $T \vdash \mathrm{Prov}_T(\ulcorner A \urcorner)$.*

Proof: Let p be the code of an actual T-proof of A. The true primitive-recursive fact $\mathrm{Prf}_T^{\mathrm{ext}}(p, \ulcorner A \urcorner)$ is represented positively in $\mathbf{Q}$, so $\mathbf{Q} \vdash \mathrm{Prf}_T(\overline{p}, \ulcorner A \urcorner)$. Since $T \supseteq \mathbf{Q}$, existential introduction gives $T \vdash \mathrm{Prov}_T(\ulcorner A \urcorner)$. □

The converse fails: from $T \nvdash A$ one cannot infer $T \vdash \neg\mathrm{Prov}_T(\ulcorner A \urcorner)$. Taking $A = \bot$ would otherwise contradict the Second Incompleteness Theorem.

9 The Diagonal Lemma

Self-reference enters through substitution on codes, not through an informal truth predicate.

Lemma 23 (Diagonal or fixed-point lemma)**:** *Let $T \supseteq \mathbf{Q}$, and let $\theta(x)$ have one free variable. There is a sentence G such that*

$$T \vdash G \leftrightarrow \theta(\ulcorner G \urcorner).$$

Proof: Let $\mathrm{diag}(u)$ be the primitive-recursive function that, when u codes a formula with the designated free variable, returns the code obtained by substituting the numeral $\overline{u}$ for that variable, and returns 0 otherwise. Let $D(u, v)$ strongly numeralwise represent its graph. Define

$$\delta(u) \equiv \exists v\,(D(u, v) \wedge \theta(v))$$

and let G be $\delta(\ulcorner \delta \urcorner)$. Externally,

$$\mathrm{diag}(\ulcorner \delta \urcorner) = \ulcorner \delta(\ulcorner \delta \urcorner) \urcorner = \ulcorner G \urcorner.$$

Strong numeralwise representation gives

$$\mathbf{Q} \vdash \forall v\,(D(\ulcorner \delta \urcorner, v) \leftrightarrow v = \ulcorner G \urcorner).$$

First-order substitution therefore yields $T \vdash \delta(\ulcorner \delta \urcorner) \leftrightarrow \theta(\ulcorner G \urcorner)$, as required. □

The sentence does not literally contain its written expression; it contains a numeral denoting its Gödel number.

10 Gödel's First Incompleteness Theorem

Apply Lemma 23 to $\theta(x) \equiv \neg\mathrm{Prov}_T(x)$. There is a sentence G_T satisfying

$$T \vdash G_T \leftrightarrow \neg\mathrm{Prov}_T(\ulcorner G_T \urcorner). \tag{3}$$

10.1 Unprovability from consistency

Proposition 24: *If T is consistent, then $T \nvdash G_T$.*

Proof: If $T \vdash G_T$, Proposition 22 gives $T \vdash \mathrm{Prov}_T(\ulcorner G_T \urcorner)$, while the left-to-right direction of (3) gives its negation. This contradicts consistency. □

10.2 Unrefutability and stronger correctness assumptions

Proposition 25: *If T is ω-consistent, then $T \nvdash \neg G_T$.*

Proof: Suppose $T \vdash \neg G_T$. By classical logic and (3), $T \vdash \mathrm{Prov}_T(\ulcorner G_T \urcorner)$, a Σ_1 sentence $\exists p\, \mathrm{Prf}_T(p, \ulcorner G_T \urcorner)$. Since ω-consistency implies consistency, Proposition 24 shows that no standard number codes a proof of G_T. For every $n \in \mathbb{N}$, therefore, the corresponding primitive-recursive instance is false and $\mathbf{Q}$ proves $\neg\mathrm{Prf}_T(\overline{n}, \ulcorner G_T \urcorner)$. Thus T proves an existential formula while refuting every numeral instance, contrary to ω-consistency. □

The same conclusion follows from 1-consistency: under consistency, $\mathrm{Prov}_T(\ulcorner G_T \urcorner)$ is a false Σ_1 sentence.

Theorem 26 (Gödel's original form)**:** *If $T \supseteq \mathbf{Q}$ is c.e. and ω-consistent, then G_T is undecidable in T.*

Corollary 27 (Truth of the standard Gödel sentence)**:** *If $T \supseteq \mathbf{Q}$ is consistent, then the standard sentence G_T constructed above is true in $\mathbb{N}$, although a consistent theory that is not 1-consistent may prove the false sentence $\neg G_T$.*

Proof: Consistency gives $T \nvdash G_T$, so no standard proof code exists and $\mathbb{N} \vDash \neg\mathrm{Prov}_T(\ulcorner G_T \urcorner)$. The fixed-point equivalence is provable in $\mathbf{Q}$, and $\mathbf{Q}$ is sound in the standard model; hence that equivalence holds in $\mathbb{N}$. Therefore $\mathbb{N} \vDash G_T$. □

Ordinary consistency is enough for truth and unprovability of the standard Gödel sentence; a stronger correctness assumption is needed only to prevent the theory from proving its false negation.

11 Rosser's Strengthening

Let $\mathrm{neg}(x)$ be the primitive-recursive function returning the code of the negation when x is a sentence code and 0 otherwise. Let $N(x, w)$ strongly numeralwise represent its graph. Define

$$\mathrm{Prov}_T^R(x) \equiv \exists y\, \exists w\, (N(x, w) \wedge \mathrm{Prf}_T(y, x) \wedge \forall z \leq y\, \neg\mathrm{Prf}_T(z, w)).$$

Apply the diagonal lemma to $\neg\mathrm{Prov}_T^R(x)$. There is a sentence R_T such that

$$T \vdash R_T \leftrightarrow \neg\mathrm{Prov}_T^R(\ulcorner R_T \urcorner). \tag{4}$$

Theorem 28 (Rosser's Incompleteness Theorem)**:** *If $T \supseteq \mathbf{Q}$ is consistent and c.e., then $T \nvdash R_T$ and $T \nvdash \neg R_T$.*

Proof: Assume first that $T \vdash R_T$, and let p code such a proof. Consistency implies that no number codes a proof of $\neg R_T$. In particular, every standard $z \leq p$ fails to do so. Since the bound is a fixed numeral, Lemma 18 and numeralwise correctness give

$$T \vdash \mathrm{Prf}_T(\overline{p}, \ulcorner R_T \urcorner) \wedge \forall z \leq \overline{p}\, \neg \mathrm{Prf}_T(z, \ulcorner \neg R_T \urcorner).$$

Together with the represented negation code, this yields $T \vdash \mathrm{Prov}_T^R(\ulcorner R_T \urcorner)$, contradicting (4) and the assumed proof of R_T. Thus $T \nvdash R_T$.

Now assume $T \vdash \neg R_T$, and let q code such a proof. From the first half, every standard $y < q$ fails to code a proof of R_T, so

$$T \vdash \forall y < \overline{q}\, \neg \mathrm{Prf}_T(y, \ulcorner R_T \urcorner).$$

Also $T \vdash \mathrm{Prf}_T(\overline{q}, \ulcorner \neg R_T \urcorner)$ and

$$T \vdash \forall w\, (N(\ulcorner R_T \urcorner, w) \leftrightarrow w = \ulcorner \neg R_T \urcorner).$$

For a proposed Rosser witness $y < \overline{q}$, the positive proof component fails. For $\overline{q} \leq y$, the value $z = \overline{q}$ refutes the bounded claim that no proof of the negation has code at most y. By the fixed-numeral comparison clause of Lemma 18, $\mathbf{Q} \vdash y < \overline{q} \vee \overline{q} \leq y$. The two cases therefore give $T \vdash \neg \mathrm{Prov}_T^R(\ulcorner R_T \urcorner)$. On the other hand, classical logic applied to (4) gives $T \vdash \neg R_T \to \mathrm{Prov}_T^R(\ulcorner R_T \urcorner)$. Together with the assumed proof of $\neg R_T$, this contradicts consistency. Hence $T \nvdash \neg R_T$. □

Rosser provability is designed for the First Theorem. It should not automatically replace the canonical provability predicate in the Second Theorem: Rosser predicates can fail familiar derivability properties, and their associated consistency statements can behave differently [7,18].

12 The Hilbert–Bernays–Löb Derivability Conditions

For the remainder, let T be a c.e. extension of $I\Sigma_1$ equipped with the canonical proof predicate of Definition 20. For sentences A, B, the conditions are

(D1) $T \vdash A \implies T \vdash \mathrm{Prov}_T(\ulcorner A \urcorner)$,

(D2) $T \vdash \mathrm{Prov}_T(\ulcorner A \to B \urcorner) \to (\mathrm{Prov}_T(\ulcorner A \urcorner) \to \mathrm{Prov}_T(\ulcorner B \urcorner))$,

(D3) $T \vdash \mathrm{Prov}_T(\ulcorner A \urcorner) \to \mathrm{Prov}_T(\ulcorner \mathrm{Prov}_T(\ulcorner A \urcorner) \urcorner)$.

The numbering varies in the literature; the content is standard [8,11].

12.1 A representative Σ_1-induction invariant

The proof transformations below process a coded finite sequence. To make the syntactic complexity of the induction explicit, fix such a transformation F. Let

$$\mathrm{PrefixOK}_F(k, \vec{a}, q, h)$$

be the bounded relation asserting that h is a correct computation trace for the first k transformation steps on input $\vec{a}$, that q is the corresponding output prefix, that every output line has a packed local verification witness, and that each output line corresponds to the intended input line or appended inference. All quantifiers are bounded by components of q, h. The induction formula is

$$\mathrm{Inv}_F(k, \vec{a}) \equiv \exists q\, \exists h\, \mathrm{PrefixOK}_F(k, \vec{a}, q, h), \tag{5}$$

which is Σ_1. The base witness is the empty output trace. The step extends the trace by one locally verified line, and the existential witnesses are the new prefix and extended trace. Thus $I\Sigma_1$ may legitimately apply Σ_1-induction to (5). The formalized modus-ponens, proof-lifting, bounded-universal, and existential-introduction transformations are instances of this template.

12.2 Condition (D1)

Condition (D1) is Proposition 22. It is a metatheoretic rule: an actual proof supplies a numeral witnessing the internal provability formula.

12.3 Condition (D2): formalized modus ponens

There is a primitive-recursive function $\mathrm{mp}(u, v)$ that concatenates a proof coded by u of $A \to B$ with a proof coded by v of A, shifts the predecessor indices in the second proof by the first proof's length, and appends B by modus ponens. Let $M_{\mathrm{mp}}(u, v, w)$ be its canonical graph formula.

Proposition 29 (Formalized modus ponens)**:** *For all sentences A, B,*

$$T \vdash \mathrm{Prov}_T(\ulcorner A \to B \urcorner) \to (\mathrm{Prov}_T(\ulcorner A \urcorner) \to \mathrm{Prov}_T(\ulcorner B \urcorner)).$$

Proof: The exact checker verifies locally that copied lines remain valid, shifted references point to the corresponding earlier lines, and the final line satisfies the modus-ponens clause. Using the invariant (5), $I\Sigma_1$ proves

$$\forall u\, \forall v\, (\mathrm{Prf}_T(u, \ulcorner A \to B \urcorner) \wedge \mathrm{Prf}_T(v, \ulcorner A \urcorner) \to \exists w\, [M_{\mathrm{mp}}(u, v, w) \wedge \mathrm{Prf}_T(w, \ulcorner B \urcorner)]).$$

Existentially choosing proof witnesses for the two antecedent provability statements and then introducing w yields (D2). □

12.4 Uniform proof synthesis for bounded formulas

For a fixed formula $\delta(\vec{x})$, let $\mathrm{inst}_\delta(\vec{a})$ be the primitive-recursive function returning the code of the closed sentence obtained by replacing each x_i with $\overline{a_i}$. Its canonical graph is denoted $I_\delta(\vec{a}, s)$.

Lemma 30 (Uniform proof synthesis for Δ_0 formulas; detailed construction schema)**:** *For every fixed Δ_0 formula $\delta(\vec{x})$, there is a primitive-recursive proof builder $b_\delta(\vec{a})$, with canonical graph $B_\delta(\vec{a}, p)$, such that $I\Sigma_1$ proves*

$$\forall \vec{a}\,(\delta(\vec{a}) \to \exists s\,\exists p\,[I_\delta(\vec{a}, s) \wedge B_\delta(\vec{a}, p) \wedge \mathrm{Prf}_{\mathbf{Q}}(p, s)]). \tag{6}$$

Proof: Fix a primitive-recursive normalization map nnf_δ producing a bounded negation-normal-form formula δ^{NNF}, together with fixed Hilbert derivations of the universal closures of $\delta \leftrightarrow \delta^{\mathrm{NNF}}$. The builder first proves the appropriate closed δ^{NNF}-instance and then appends the corresponding instance of $\delta^{\mathrm{NNF}} \to \delta$. Its last formula is therefore the original instance whose code is specified by I_δ, not merely the normalized one.

The builder is a total numerical function: on every input it first evaluates the relevant closed bounded formula, which is decidable by Lemma 7; when the formula is true it returns the proof code constructed below, and when it is false it returns a fixed dummy code, say 0. The claim (6) asserts correctness only under the antecedent $\delta(\vec{a})$, so the dummy branch never affects the asserted implication.

The construction in the true case is structural recursion on δ^{NNF}. For an atomic equality, evaluate the closed numeral terms, construct **Q**-proofs of their values from the defining equations for addition and multiplication, and join the calculations by equality reasoning. A negative atomic equality is handled by the fixed decision procedure for distinct numerals; inequalities are treated through their bounded-existential definition. For a conjunction, the builder recurses on both conjuncts and appends the conjunction-introduction derivation. For a disjunction, the builder evaluates the disjuncts in order, selects the first true one, recurses on it, and appends the corresponding disjunction-introduction derivation; if neither is true the formula is false and the dummy branch applies.

For $\exists z < t(\vec{x})\,\eta(\vec{x}, z)$, the builder computes the numeral value m of $t(\vec{a})$ and performs bounded search for the least $j < m$ with $\eta(\vec{a}, \overline{j})$ true. When such a j exists, the inductively supplied builder proves that η-instance, and a fixed transformer appends a proof of the numeral bound $\overline{j} < \overline{m}$ and existential introduction; when no such j exists the formula is false and the dummy branch applies. For $\forall z < t(\vec{x})\,\eta(\vec{x}, z)$, compute the numeral value m of $t(\vec{a})$. Primitive recursion on m concatenates the proofs of $\eta(\vec{a}, 0), \ldots, \eta(\vec{a}, m-1)$ when all are true; if some instance is false the formula is false and the dummy branch applies. A second constructor appends the finite numeral-classification proof of Lemma 18 and converts the conjunction into the bounded universal.

Correctness is expressed by invariant (5): its trace records the processed index, the output proof prefix, and packed verification traces for every generated line. The invariant is Σ_1, so $I\Sigma_1$ proves it by Σ_1-induction on m. Every recursive clause outputs an exact proof sequence accepted by the checker of Theorem 11. This is a detailed construction schema rather than a printed numerical code for every fixed Hilbert derivation. Lemma 21 supplies uniform totality and functionality of the builders, establishing (6). □

Lemma 31 (Uniform lifting of base-theory proofs)**:** *Let $T \supseteq \mathbf{Q}$ have the normalized presentation of Definition 20. There is a primitive-recursive function* $\mathrm{lift}_{\mathbf{Q},T}(p)$*, with canonical graph $L_{\mathbf{Q},T}(p, q)$, such that $I\Sigma_1$ proves*

$$\forall p\, \forall s\, (\mathrm{Prf}_{\mathbf{Q}}(p, s) \to \exists q\, [L_{\mathbf{Q},T}(p, q) \wedge \mathrm{Prf}_T(q, s)]). \tag{7}$$

Proof: Traverse the proof line by line. Logical-axiom and inference lines are copied. A line tagged as a **Q**-axiom is retagged as a base-axiom line accepted by the normalized T-checker. Formula codes and predecessor indices are unchanged. If a different presentation places the finite **Q**-axioms in the c.e. enumeration, insert fixed stage certificates for those seven axioms. The invariant (5) asserts that the processed output prefix is a valid T-proof prefix with the same formula sequence. Its local step is immediate from the exact line checker, and $I\Sigma_1$ proves the result by Σ_1-induction on the input length. □

12.5 Formalized Σ_1-completeness

Lemma 32 (Formalized Σ_1-completeness)**:** *For every Σ_1 sentence σ,*

$$T \vdash \sigma \to \mathrm{Prov}_T(\ulcorner \sigma \urcorner).$$

Proof: Write $\sigma \equiv \exists x\, \delta(x)$ with $\delta \in \Delta_0$, coding tuples if necessary. Lemmas 30 and 31 give

$$I\Sigma_1 \vdash \delta(a) \to \exists s\, \exists p\, [I_\delta(a, s) \wedge \mathrm{Prf}_T(p, s)]. \tag{8}$$

There is a primitive-recursive transformer $\mathrm{ex}_\sigma(a, s, p)$ that, when s codes $\delta(a)$, appends the existential-introduction derivation yielding σ. Its canonical graph $E_\sigma(a, s, p, q)$ satisfies, by the trace-correctness lemma and the invariant (5),

$$I\Sigma_1 \vdash I_\delta(a, s) \wedge \mathrm{Prf}_T(p, s) \to \exists q\, [E_\sigma(a, s, p, q) \wedge \mathrm{Prf}_T(q, \ulcorner \sigma \urcorner)]. \tag{9}$$

Combining (8) and (9), then applying first-order existential reasoning, yields

$$I\Sigma_1 \vdash (\exists a\, \delta(a)) \to \mathrm{Prov}_T(\ulcorner \sigma \urcorner).$$

Since $T \supseteq I\Sigma_1$, the same implication is provable in T. □

This is a separate theorem for each fixed Σ_1 sentence, not a single internal truth predicate for all externally true Σ_1 sentences.

12.6 Condition (D3): provable provability

Proposition 33 (Condition (D3))**:** *For every sentence A,*

$$T \vdash \mathrm{Prov}_T(\ulcorner A \urcorner) \to \mathrm{Prov}_T(\ulcorner \mathrm{Prov}_T(\ulcorner A \urcorner) \urcorner).$$

Proof: The closed formula $\mathrm{Prov}_T(\ulcorner A \urcorner)$ is a Σ_1 sentence. Apply Lemma 32 to that sentence. □

13 Löb's Theorem and the Second Incompleteness Theorem

13.1 Löb's theorem

Theorem 34 (Löb)**:** *Assume* Prov_T *satisfies (D1)–(D3). If*

$$T \vdash \mathrm{Prov}_T(\ulcorner A \urcorner) \to A,$$

then $T \vdash A$.

Proof: By the diagonal lemma, choose L such that

$$T \vdash L \leftrightarrow (\mathrm{Prov}_T(\ulcorner L \urcorner) \to A). \tag{10}$$

From the left-to-right direction of (10), (D1), and (D2),

$$T \vdash \mathrm{Prov}_T(\ulcorner L \urcorner) \to \mathrm{Prov}_T(\ulcorner \mathrm{Prov}_T(\ulcorner L \urcorner) \to A \urcorner).$$

A second use of (D2), together with (D3), gives

$$T \vdash \mathrm{Prov}_T(\ulcorner L \urcorner) \to \mathrm{Prov}_T(\ulcorner A \urcorner).$$

By the hypothesis, $T \vdash \mathrm{Prov}_T(\ulcorner L \urcorner) \to A$. The right-to-left direction of (10) yields $T \vdash L$. By (D1), $T \vdash \mathrm{Prov}_T(\ulcorner L \urcorner)$, and therefore $T \vdash A$. □

13.2 The standard consistency sentence

Define

$$\mathrm{Con}(T) \equiv \neg \mathrm{Prov}_T(\ulcorner \bot \urcorner).$$

In the standard model, this says that no natural number codes a T-proof of contradiction. The adjective "standard" matters: a formula can agree with provability on every standard numeral pair while failing the uniform properties needed above. The theorem concerns the canonical arithmetization of the chosen proof calculus.

Theorem 35 (Second Incompleteness Theorem)**:** *If* T *is consistent and its canonical provability predicate satisfies (D1)–(D3), then*

$$T \nvdash \mathrm{Con}(T).$$

Proof: Suppose $T \vdash \mathrm{Con}(T)$. Classical logic rewrites this as

$T \vdash \mathrm{Prov}_T(\ulcorner \bot \urcorner) \to \bot$.

Apply Löb's theorem with $A = \bot$. Then $T \vdash \bot$, contradicting consistency. □

The theorem does not say that no consistency proof for T exists. It says that a consistent T cannot prove its own canonical arithmetized consistency statement. Stronger theories may prove $\mathrm{Con}(T)$, though the same question then arises for the stronger theory. Gentzen's consistency proof for **PA**, for example, uses transfinite induction beyond the induction formalized in **PA** itself [4].

14 Truth, Provability, and Tarski's Theorem

Provability in a fixed c.e. theory is arithmetically definable. Truth in the standard model is different.

Definition 36 (Arithmetical truth set)**:** *Fix the Gödel numbering used above. The complete first-order truth set of the standard model is*

$\mathrm{Th}(\mathbb{N}) = \{\ulcorner \varphi \urcorner : \varphi \text{ is an } \mathcal{A}\text{-sentence and } \mathbb{N} \vDash \varphi\}$.

A set $X \subseteq \mathbb{N}$ is arithmetically definable in $\mathbb{N}$ *if an $\mathcal{A}$-formula $\xi(x)$ satisfies $n \in X \iff \mathbb{N} \vDash \xi(\overline{n})$ for every $n \in \mathbb{N}$.*

Theorem 37 (Tarski's undefinability theorem for arithmetic)**:** *There is no arithmetical formula* $\mathrm{Tr}(x)$ *defining* $\mathrm{Th}(\mathbb{N})$ *in $\mathbb{N}$ [19]. Equivalently, no $\mathcal{A}$-formula satisfies, for every arithmetical sentence φ,*

$$\mathbb{N} \vDash \mathrm{Tr}(\ulcorner \varphi \urcorner) \iff \mathbb{N} \vDash \varphi. \tag{11}$$

Equivalently again, no arithmetical predicate satisfies all semantic Tarski biconditionals $\mathbb{N} \vDash \mathrm{Tr}(\ulcorner \varphi \urcorner) \leftrightarrow \varphi$ simultaneously.

Proof: Assume such a formula exists. Apply the diagonal lemma over **Q** to $\neg\mathrm{Tr}(x)$. There is a sentence λ such that

$\mathbf{Q} \vdash \lambda \leftrightarrow \neg\mathrm{Tr}(\ulcorner \lambda \urcorner)$.

Since $\mathbb{N} \vDash \mathbf{Q}$,

$$\mathbb{N} \vDash \lambda \leftrightarrow \neg\mathrm{Tr}(\ulcorner \lambda \urcorner). \tag{12}$$

But (11) gives $\mathbb{N} \vDash \mathrm{Tr}(\ulcorner \lambda \urcorner)$ exactly when $\mathbb{N} \vDash \lambda$. Combining this with (12) yields a contradiction. □

Exact definability of $\mathrm{Th}(\mathbb{N})$ and the full biconditional scheme are equivalent because $\mathrm{Sent}(x)$ is arithmetically definable: if a formula satisfied every biconditional but behaved arbitrarily on nonsentence codes, $\mathrm{Sent}(x) \wedge \mathrm{Tr}(x)$ would define exactly $\mathrm{Th}(\mathbb{N})$.

The theorem is semantic and concerns the complete truth set of the standard model. It does not say that truth cannot be defined in a richer metalanguage: set theory defines satisfaction for $\mathbb{N}$. Nor does it rule out partial truth predicates. For every fixed level of the arithmetical hierarchy, suitable formulas define truth for formulas of that bounded complexity. What fails is one arithmetical predicate capturing truth for all arithmetical sentences at once.

The following conclusions should be separated:

(i) If $T \supseteq \mathbf{Q}$ is consistent and c.e., the standard Gödel sentence is true in $\mathbb{N}$ and unprovable in T.
(ii) A merely consistent T may prove the false sentence $\neg G_T$; 1-consistency or ω-consistency prevents this.
(iii) Rosser's sentence is undecidable from ordinary consistency, although its informal semantic reading is less direct than that of G_T.
(iv) No consistent effective theory extending $\mathbf{Q}$ captures every first-order arithmetical truth.

The phrase "true but unprovable" is legitimate only after the model and theory have been specified.

15 Natural Independence Phenomena

Gödel sentences are engineered from a proof predicate, but independence also occurs in recognizable mathematics.

- The Continuum Hypothesis is independent of **ZFC** relative to the consistency of **ZFC**. Gödel established the relative consistency of the Generalized Continuum Hypothesis using the constructible universe, and Cohen established the relative consistency of its negation by forcing [2,6].
- The Paris–Harrington principle is a finite Ramsey-type statement independent of **PA** [13].
- Goodstein's theorem is a concrete number-theoretic statement true in $\mathbb{N}$ but unprovable in **PA** [9].
- Shelah proved that the Whitehead problem for abelian groups is independent of **ZFC** in the relative-consistency sense [16].

These results use different methods; forcing, ordinal analysis, and finite combinatorics should not be collapsed into one technique. Their common lesson is narrower: formal independence is not confined to sentences explicitly mentioning proofs.

16 Connections to Computability and Computer Science

Incompleteness and undecidability are related but distinct. Turing's undecidability argument implies that no algorithm decides, for every program and input, whether that computation halts [20]. A complete, sound, effective decision method for sufficiently rich arithmetic would decide corresponding encodings of computation and conflict with such undecidability. Conversely, arithmetized computation gives direct routes to incompleteness.

The consequences for computer science are structural rather than defeatist.

- *Automated theorem proving.* No prover whose theorem set is one fixed consistent c.e. extension of sufficient arithmetic decides every arithmetical truth.
- *Proof assistants.* Lean, Coq, Isabelle, and related systems can check enormous formal developments, but confidence in a result also depends on metatheoretic assumptions about the kernel, compiler, hardware, and surrounding implementation.
- *Program verification.* General semantic properties of arbitrary programs are subject to undecidability, so complete verification methods require restrictions on languages, specifications, or program classes.
- *Artificial intelligence.* Gödelian arguments do not by themselves prove that human reasoning is noncomputable. Anti-mechanist conclusions require additional assumptions about a human reasoner's consistency or soundness and about representation by one fixed c.e. theory.

Lucas and Penrose developed influential anti-mechanist arguments, but their conclusions remain controversial because a human mathematician is neither evidently sound nor evidently captured by a single recursively axiomatized theory [3,12,14].

17 Philosophical Significance and Limits

17.1 What the theorems establish

Every consistent c.e. extension of sufficient arithmetic is syntactically incomplete. A consistent theory whose canonical provability predicate satisfies the derivability conditions cannot prove its own standard consistency statement. For every consistent c.e. $T \supseteq \mathbf{Q}$, provability in T does not exhaust truth in the standard model. Foundational justification therefore cannot be supplied by one final effective theory that both encompasses ordinary arithmetic and certifies itself by its own standard methods.

17.2 What the theorems do not establish

The theorems do not show that mathematics is inconsistent, that most mathematical questions are undecidable, that formal proof is futile, or that every open conjecture is independent. They do not prove that human cognition transcends computation. The inference from "for every consistent c.e. theory there is a sentence it does not prove" to "a human can correctly see every corresponding sentence" requires a uniform and justified account of the human reasoner's soundness, which incompleteness does not provide.

Nor do the theorems prohibit relative consistency proofs. A stronger system can often prove the consistency of a weaker one. What is blocked is self-certification by a sufficiently strong consistent theory through its own canonical provability predicate.

18 Common Misconceptions

"Gödel proved that mathematics is inconsistent." False. The theorems are conditional on consistency and yield incompleteness or unprovability of a canonical consistency statement.
"The Gödel sentence is a liar paradox." False. It is an arithmetical fixed point concerning formal provability, not a semantic sentence directly asserting its own falsity.
"Ordinary consistency proves both halves of Gödel's original theorem." False. Consistency proves $T \nvdash G_T$ and the truth of the standard G_T in $\mathbb{N}$. To conclude $T \nvdash \neg G_T$ by the original construction, one uses

ω-consistency, 1-consistency, or a comparable correctness assumption. Rosser's modified sentence requires only consistency.

"Every formal system is incomplete." False. The hypotheses include effective axiomatizability and sufficient arithmetical strength. Presburger arithmetic is complete and decidable but too weak to express full multiplication; the set of all true first-order arithmetical sentences is complete but not c.e.

"Tarski proved that truth cannot be discussed mathematically." False. Tarski proved that the full truth set of $\mathbb{N}$ is not definable by a formula of the same first-order language. Truth is definable in stronger metalanguages, and partial truth predicates exist for restricted complexity classes.

"A theory cannot have any consistency proof." False. The Second Theorem concerns the theory's own canonical internal statement $\mathrm{Con}(T)$. Stronger metatheories can prove relative consistency results.

"Gödel proved that people are not machines." False as a statement of theorem. Lucas–Penrose arguments require additional premises and remain philosophically contested.

19 Limitations of the Present Exposition

This article gives a detailed paper-level construction of prime-power sequence coding, a fresh-variable-correct substitution algorithm with separate explicit primitive-recursive majorants M_{RenF} and M_{Sub}, an exact Hilbert calculus, primitive-recursive proof verification for c.e. axiom sets, numeralwise representability, explicit Σ_1 witness coding, and uniform bounded proof synthesis. It nevertheless stops short of a machine-checked formalization. In particular, Lemmas 21 and 30 are detailed construction schemas: they specify the recursive clauses, trace invariants, and proof-building strategy, but they do not print executable definitions of $\mathrm{PrefixOK}_F$, nnf_δ, every local proof constructor, or every code-growth bound. A proof assistant would require executable definitions for every syntactic category and constructor and verification of every line of every proof transformation.

The coding is chosen for transparency rather than computational efficiency, and no claim is made about optimal code growth. The use of $I\Sigma_1$ is convenient rather than minimal; weaker bases suffice in refined treatments. The natural-independence survey is illustrative, not comprehensive. Finally, the philosophical discussion evaluates direct consequences of the formal results rather than offering a general theory of mathematical understanding or machine intelligence.

The manuscript is therefore more explicit than a textbook sketch but less granular than a verified formal library. Its aim is rigorous ordinary metamathematics with the important transitions between external computation and internal arithmetic stated openly.

20 Conclusions and Future Work

Gödel's incompleteness theorems arise from a precise chain. Syntax is coded by natural numbers through prime-power sequences, tagged constructors, and primitive-recursive capture-avoiding substitution. A proof is a finite sequence of line records checked by a local primitive-recursive predicate; for c.e. axiom sets, bounded simulation of a deterministic enumerator supplies stage certificates. The representability theorem transfers these external numerical relations into arithmetic. Its primitive-recursion step uses

β-coded histories, an entry formula with explicit internal uniqueness, a finite-unfolding theorem for numeral bounds, and a packed witness history displaying the promised Σ_1 form.

The diagonal lemma produces sentences applying arithmetical predicates to their own codes. Gödel's original sentence separates the role of consistency from that of 1- or ω-consistency; Rosser's ordering of proof codes obtains syntactic undecidability from ordinary consistency. For the Second Theorem, the canonical proof predicate matters: agreement on standard numeral pairs is insufficient. Canonical trace formulas, explicit Σ_1-induction invariants, uniform proof lifting, and bounded proof synthesis in $I\Sigma_1$ yield formalized Σ_1-completeness, the derivability conditions, Löb's theorem, and unprovability of the canonical consistency statement. Tarski's theorem separately shows that no arithmetical predicate defines the complete truth set of the standard model.

The conceptual lessons are correspondingly exact. Effective axiomatizability is indispensable. Numeralwise representability in **Q** is not uniform internal totality. A proof predicate is not a provability predicate. Consistency is weaker than 1-consistency, ω-consistency, and soundness. Provability in a fixed c.e. theory is definable, while complete truth in $\mathbb{N}$ is not definable in the same language. Finally, the theorems constrain fixed formal theories; they do not by themselves establish that human thought is nonmechanical.

Promising extensions include a machine-checked implementation in Lean, Coq, or Isabelle; comparison of the exact resources required in **Q**, primitive-recursive arithmetic, elementary arithmetic, $I\Sigma_1$, and **PA**; versions for natural deduction and sequent calculi; connection with the modal provability logic GL; and detailed treatments of Goodstein sequences, Paris–Harrington, and ordinal analysis.

Author Contributions: *Aliefya Mahalva Ayn:* Conceptualization, Formal analysis, Methodology, Writing—Original Draft, and Writing—Review and Editing. The author has read and approved the final version of the manuscript for publication.

Acknowledgement: The author gratefully acknowledges Professor Ted Sider for guidance during an undergraduate independent study that led to an early draft of this article, and thanks readers who provided constructive comments on later versions.

Funding Statement: The author received no specific funding for this study.

Data Availability Statement: Not applicable. This is a theoretical study and no datasets were generated or analyzed.

Ethics Approval: Not applicable.

Use of Generative-AI tools declaration: The author declares that no Artificial Intelligence (AI) tools were used in the creation of this article.

Conflicts of Interest: The author declares no competing interests.

References

1. G. S. Boolos, J. P. Burgess, and R. C. Jeffrey, *Computability and Logic*, 5th ed., Cambridge University Press, **2007**.
2. P. J. Cohen, The independence of the continuum hypothesis, *Proc. Natl. Acad. Sci. USA* **50 (1963)** , 1143–1148. https://doi.org/10.1073/pnas.50.6.1143

3. T. Franzén, *Gödel's Theorem: An Incomplete Guide to Its Use and Abuse*, A K Peters, **2005**.
4. G. Gentzen, Die Widerspruchsfreiheit der reinen Zahlentheorie, *Math. Ann.* **112** (1936), 493–565. https://doi.org/10.1007/BF01565428
5. K. Gödel, Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme I, *Monatsh. Math. Phys.* **38** (1931), 173–198. https://doi.org/10.1007/BF01700692
6. K. Gödel, *The Consistency of the Axiom of Choice and of the Generalized Continuum-Hypothesis with the Axioms of Set Theory*, Princeton University Press, **1940**.
7. P. Hájek and P. Pudlák, *Metamathematics of First-Order Arithmetic*, Springer, **1993**. https://doi.org/10.1007/978-3-642-59103-2
8. D. Hilbert and P. Bernays, *Grundlagen der Mathematik II*, Springer, **1939**.
9. L. Kirby and J. Paris, Accessible independence results for Peano arithmetic, *Bull. London Math. Soc.* **14**(4) (1982), 285–293. https://doi.org/10.1112/blms/14.4.285
10. S. C. Kleene, *Introduction to Metamathematics*, North-Holland, **1952**.
11. M. H. Löb, Solution of a problem of Leon Henkin, *J. Symbolic Logic* **20**(2) (1955), 115–118. https://doi.org/10.2307/2266895
12. J. R. Lucas, Minds, machines and Gödel, *Philosophy* **36**(137) (1961), 112–127. https://doi.org/10.1017/S0031819100057983
13. J. Paris and L. Harrington, A mathematical incompleteness in Peano arithmetic, in J. Barwise (ed.), *Handbook of Mathematical Logic*, North-Holland, **1977**, pp. 1133–1142. https://doi.org/10.1016/S0049-237X(08)71170-9
14. R. Penrose, *The Emperor's New Mind*, Oxford University Press, **1989**.
15. J. B. Rosser, Extensions of some theorems of Gödel and Church, *J. Symbolic Logic* **1**(3) (1936), 87–91. https://doi.org/10.2307/2269028
16. S. Shelah, Infinite abelian groups, Whitehead problem and some constructions, *Israel J. Math.* **18** (1974), 243–256. https://doi.org/10.1007/BF02757281
17. P. Smith, *An Introduction to Gödel's Theorems*, 2nd ed., Cambridge University Press, **2013**.
18. R. M. Smullyan, *Gödel's Incompleteness Theorems*, Oxford University Press, **1992**.
19. A. Tarski, The concept of truth in formalized languages, in *Logic, Semantics, Metamathematics*, Clarendon Press, **1956**, pp. 152–278.
20. A. M. Turing, On computable numbers, with an application to the Entscheidungsproblem, *Proc. London Math. Soc.* **42** (1937), 230–265. https://doi.org/10.1112/plms/s2-42.1.230