10 The Unprovability of Consistency

Gödel's Second Incompleteness Theorem states that no sufficiently strong mathematical theory can prove its own consistency. In this chapter, we explore how this can be shown and what it implies for the foundations of mathematics and beyond.

10.1 The Second Incompleteness Theorem

Let's review some background. As in the previous chapter, we're going to focus on axiomatizable theories in the language of arithmetic $\mathfrak{L}_A$. A *theory* is simply a set of sentences that is closed under logical consequence. A *(recursively) axiomatizable* theory is one for which there is a (recursively) decidable set of axioms from which all and only the members of the theory are deducible.

By coding $\mathfrak{L}_A$-strings as numbers, we can use the language of arithmetic to reason about its own syntax. We use '$\#[A]$' to denote the code ("Gödel number") of an $\mathfrak{L}_A$-string A, and '$\ulcorner A \urcorner$' to denote the $\mathfrak{L}_A$-numeral for $\#[A]$, relative to some fixed coding scheme. (The details of the scheme don't matter, as long as it is effective.)

Given a recursively axiomatizable theory T, let Prf_T be the relation that holds between numbers n and m iff n codes a deduction of the sentence coded by m from some recursively decidable set of axioms for T. This relation is recursive. In Chapter 8 we proved that all recursive functions and relations are representable in any moderately strong theory of arithmetic – in particular, in any extension of Q. That is, if T is at least as strong as Q then there is an $\mathfrak{L}_A$-formula $\text{PRF}_T(x, y)$ such that

(i) if $\text{Prf}_T(n, m)$ holds, then $\vdash_T \text{PRF}_T(\ulcorner n \urcorner, \ulcorner m \urcorner)$;
(ii) if $\text{Prf}_T(n, m)$ doesn't hold, then $\vdash_T \neg\text{PRF}_T(\ulcorner n \urcorner, \ulcorner m \urcorner)$.

If T is sound, the formula $\text{PRF}_T(x, y)$ also defines Prf_T in $\mathfrak{L}_A$, so that

$$\mathfrak{A} \Vdash \text{PRF}_T(\ulcorner n \urcorner, \ulcorner m \urcorner) \text{ iff } \text{Prf}_T(n, m).$$

From $\text{PRF}_T(x, y)$, we can define another formula $\text{PROV}_T(x)$ as $\exists y\, \text{PRF}_T(y, x)$. Informally, $\text{PROV}_T(x)$ says that the sentence with Gödel number x is provable in T.

Now remember that a theory T is *consistent* if it doesn't prove a contradiction. This can be spelled out in multiple ways:

(1) There is no sentence A such that $\vdash_T A$ and $\vdash_T \neg A$;
(2) $\nvdash_T \bot$;
(3) There is some sentence A such that $\nvdash_T A$.

In classical logic, all three conditions are equivalent. Let's focus on (2), since it is the shortest. If we have an $\mathfrak{L}_A$-formula $\text{PROV}_T(x)$ that expresses provability in T, we can find another $\mathfrak{L}_A$ formula that expresses the consistency of T: $\neg\text{PROV}_T(\ulcorner\bot\urcorner)$.

(Officially, '$\bot$' is not part of $\mathfrak{L}_A$: in Chapter 1, I suggested that it abbreviates '$\neg(p \to p)$', but we also don't have sentence letters in $\mathfrak{L}_A$. Let's say that $\bot$ is the sentence '$\neg(0 = 0)$'. Since $0 = 0$ is provable in first-order logic, $\bot$ is refutable in first-order logic, which is all that matters.)

Now let T be some recursively axiomatizable extension of Q. Since the consistency of T can be expressed in T's language, we might wonder whether T can prove it: can an arithmetical theory prove its own consistency? At the end of the 1931 paper in which he proved the First Incompleteness Theorem, Gödel gave an answer: he claimed that no sufficiently strong, recursively axiomatizable, and consistent theory can prove its own consistency. This is Gödel's Second Incompleteness Theorem.

Gödel also outlined a proof of this claim. It goes as follows.

Remember that in his proof of the First Incompleteness Theorem, Gödel used the (Syntactic) Diagonal Lemma to construct a "Gödel sentence" G such that

$$\vdash_T G \leftrightarrow \neg\text{PROV}_T(\ulcorner G \urcorner). \tag{1}$$

He then showed that *if T is consistent then T can't prove G.* He also showed that if T is ω-consistent then it can't prove $\neg G$. But let's focus on the first result. Its proof required no advanced mathematics. (No transfinite ordinals or the Axiom of Choice or anything like that.) It can be carried out in any moderately strong theory that can reason about recursivity, representability, and the syntax of T. Suppose it can be carried out in T itself. Then the following sentence is provable in T:

$$\neg\text{PROV}_T(\ulcorner\bot\urcorner) \to \neg\text{PROV}_T(\ulcorner G \urcorner). \tag{2}$$

This says in $\mathfrak{L}_A$ that if T is consistent then T can't prove G.

Now suppose that T can prove its own consistency: it can prove $\neg\text{PROV}_T(\ulcorner\bot\urcorner)$. By Modus Ponens and (2), T can infer $\neg\text{PROV}_T(\ulcorner G \urcorner)$. By (1), it can then infer G. But we've assumed that T is a recursively axiomatizable extension of Q. And we know from Gödel's First Incompleteness Theorem that no recursively axiomatizable and *consistent* extension of Q can prove G. So T must be inconsistent.

In sum, if T is a consistent and recursively axiomatizable extension of Q, and (2) is provable in T, then T can't prove its own consistency.

It turns out that (2) isn't provable in Q. But it is provable in the standard axiomatization of arithmetic, Peano Arithmetic (PA). The Second Incompleteness Theorem therefore implies that PA can't prove its own consistency (assuming it is consistent).

Before we investigate why (2) is provable in PA, let's think about the significance of the result. Why should we care whether PA can prove its own consistency?

Note that any *inconsistent* theory can trivially prove its own consistency (as long as this is expressible in its language): an inconsistent theory can prove everything. Even if a consistent theory could prove its own consistency, this would therefore provide no reason to think that the theory really is consistent.

The significance of the Second Incompleteness Theorem comes from what it implies about the powers of theories to prove the consistency of *other* theories. Take a theory like ZFC in which most of mathematics can be formalized. Hilbert had hoped that one could prove the consistency of such theories in a much weaker, "finitary" theory that studies deductions as finite syntactic objects. Gödel's Second Incompleteness Theorem implies that this can't be done. ZFC is certainly strong enough to prove (2). By Gödel's Theorem, it follows that ZFC can't prove its own consistency. But then *no weaker theory* can prove the consistency of ZFC either: if something isn't even provable in ZFC, it can't be provable in, say, PA or Q – for anything that's provable in PA or Q is also provable (under a suitable translation, see Section 4.3) in ZFC.

Exercise 10.1 The Second Incompleteness Theorem allows that the consistency of ZFC could be proved in a theory stronger than ZFC. Explain why this would hardly reassure skeptics who doubt the consistency of ZFC.

To complete the proof of the Second Incompleteness Theorem, we'd need to show that sufficiently powerful theories like PA prove (2). Fortunately, this doesn't require formalizing the entire proof of the First Incompleteness Theorem. It is enough to show that the formula $\text{PROV}_T(x)$ satisfies some basic conditions. These conditions are known as the *Hilbert-Bernays-Löb provability conditions*, because they were first formulated explicitly in a 1939 textbook by Hilbert and Bernays and later streamlined by Martin Löb. They are as follows.

P1 If $\vdash_T A$, then $\vdash_T \text{PROV}_T(\ulcorner A \urcorner)$.

P2 $\vdash_T \text{PROV}_T(\ulcorner A \to B \urcorner) \to (\text{PROV}_T(\ulcorner A \urcorner) \to \text{PROV}_T(\ulcorner B \urcorner))$.

P3 $\vdash_T \text{PROV}_T(\ulcorner A \urcorner) \to \text{PROV}_T(\ulcorner \text{PROV}_T(\ulcorner A \urcorner) \urcorner)$.

We'll see below how (2) can be derived from P1–P3. First, let's examine what it takes to show that, say, $\text{PROV}_{\text{PA}}(x)$ satisfies the three conditions.

P1 is easy. Assume that A is provable in PA. Then there is number n that codes a deduction of A from the axioms of PA. Since $\text{PRF}_{\text{PA}}(x, y)$ represents Prf_{PA} in PA, we have $\vdash_{\text{PA}} \text{PRF}_{\text{PA}}(\overline{n}, \ulcorner A \urcorner)$. So we also have $\vdash_{\text{PA}} \exists y\, \text{PRF}_{\text{PA}}(y, \ulcorner A \urcorner)$ and therefore $\vdash_{\text{PA}} \text{PROV}_{\text{PA}}(\ulcorner A \urcorner)$. So $\text{PROV}_{\text{PA}}(x)$ satisfies P1.

The argument for P2 isn't much harder. Suppose we have a proof of $A \to B$ and a proof of A. From these, we can construct a proof of B by concatenating the two proofs and adding B as a final line (applying Modus Ponens). All this can be formalized in PA, showing that if there are numbers m and n that code proofs of A and $A \to B$, then there is a number k that codes a proof of B. Which is what P2 says.

The argument for P3 is more involved. It essentially requires formalizing within PA the proof that PROV_{PA} satisfies P1. This takes about a dozen pages of tedious tinkering. I'll spare us the details. (You can find them, for example, in George Boolos, *The Logic of Provability*, 1993.) Let's just accept that $\text{PROV}_{\text{PA}}(x)$ satisfies P1–P3.

Exercise 10.2 Using the soundness of PA, show that $\text{PROV}_{\text{PA}}(x)$ also satisfies the following condition:

CNec If $\vdash_T \text{PROV}_T(\ulcorner A \urcorner)$ then $\vdash_T A$.

Exercise 10.3 Explain why it doesn't follow from the previous exercise that $\text{PROV}_{\text{PA}}(x)$ satisfies this condition:

Ref $\vdash_T \text{PROV}_T(\ulcorner A \urcorner) \to A$.

We can now finish the proof of the Second Incompleteness Theorem, by deriving (2) from P1–P3. To remove clutter, I'll abbreviate $\text{PROV}_T(\ulcorner A \urcorner)$ as $\Box A$. This is a little misleading because it suggests that the sentence A occurs in $\Box A$, while in fact only its Gödel numeral does. But it will make the proof more readable.

With this convention, P1–P3 can be written as follows:

P1 If $\vdash_T A$, then $\vdash_T \Box A$.

P2 $\vdash_T \Box(A \to B) \to (\Box A \to \Box B)$.

P3 $\vdash_T \Box A \to \Box\Box A$.

Theorem 10.1: Gödel's Second Incompleteness Theorem

If T is a consistent and recursively axiomatizable theory in which diag is representable, and $\mathrm{PROV}_T(x)$ is a formula that satisfies P1–P3, then T cannot prove $\neg\mathrm{PROV}_T(\ulcorner\bot\urcorner)$.

Proof. Applying the (Syntactic) Diagonal Lemma to the formula $\neg\mathrm{PROV}_T(x)$, we get a sentence G such that $\vdash_T G \leftrightarrow \neg\mathrm{PROV}_T(\ulcorner G\urcorner)$. Using the fact that $\mathrm{PROV}_T(x)$ satisfies P1–P3, we now reason as follows.

1.	$\vdash_T G \leftrightarrow \neg\Box G$	(Diagonal Lemma)
2.	$\vdash_T G \to \neg\Box G$	(from 1)
3.	$\vdash_T G \to (\Box G \to \bot)$	(from 2)
4.	$\vdash_T \Box(G \to (\Box G \to \bot))$	(from 3 by P1)
5.	$\vdash_T \Box G \to \Box(\Box G \to \bot)$	(from 4 by P2)
6.	$\vdash_T \Box G \to (\Box\Box G \to \Box\bot)$	(from 5 by P2)
7.	$\vdash_T \Box G \to \Box\Box G$	(P3)
8.	$\vdash_T \Box G \to \Box\bot$	(from 6 and 7)
9.	$\vdash_T \neg\Box\bot \to \neg\Box G$	(from 8)

Line 9 is (2) in the box notation. From here, the argument continues as explained above:

10.	$\vdash_T \neg\Box\bot$	(Assumption)
11.	$\vdash_T \neg\Box G$	(from 9 and 10)
12.	$\vdash_T G$	(from 1 and 11)
13.	$\vdash_T \Box G$	(from 12 by P1)

This shows that if T proves its own consistency (line 10) then it is inconsistent: it proves both $\mathrm{PROV}_T(\ulcorner G\urcorner)$ (line 13) and $\neg\mathrm{PROV}_T(\ulcorner G\urcorner)$ (line 11). By contraposition: if

T is consistent, it can't prove its own consistency. □

Why is this an *incompleteness* theorem? Because it shows that a certain sentence, $\neg\text{PROV}_T(\ulcorner\bot\urcorner)$, is unprovable in any sufficiently strong and consistent theory T, even though it is true. In a way, this form of incompleteness is more striking than the one established by Gödel's First Theorem. Here is how Gödel put it in a 1951 lecture:

> [The second theorem] makes the incompletability of mathematics particularly evident. For, it makes it impossible that someone should set up a certain well-defined system of axioms and rules and consistently make the following assertion about it: All of these axioms and rules I perceive (with mathematical certitude) to be correct, and moreover I believe that they contain all of mathematics. If someone makes such a statement he contradicts himself. For if he perceives the axioms under consideration to be correct, he also perceives (with the same certainty) that they are consistent. Hence he has a mathematical insight not derivable from his axioms.

Take the standard axiomatization of set theory, ZFC. Suppose we "perceive" that ZFC is sound, and therefore consistent. If this "perception" is correct, it goes beyond what ZFC can prove, although the consistency part can be expressed in the language of ZFC: we perceive the truth of $\neg\text{PROV}_{\text{ZFC}}(\ulcorner\bot\urcorner)$. We might propose a strengthened version of ZFC, with $\neg\text{PROV}_{\text{ZFC}}(\ulcorner\bot\urcorner)$ as a new axiom. Call this theory ZFC$^+$. The Second Incompleteness Theorem still applies: ZFC$^+$ can't prove its own consistency (if it is consistent). But if ZFC is sound, then so is ZFC$^+$. Our "perception" of the soundness of ZFC therefore allows us to strengthen ZFC$^+$ by adding another consistency statement, $\neg\text{PROV}_{\text{ZFC}^+}(\ulcorner\bot\urcorner)$. Call this theory ZFC^{++}. The Second Incompleteness Theorem still applies .... In this way, our perception of the soundness of ZFC allows us to construct an infinite sequence of ever stronger theories, each of which must be sound. We might even propose a theory ZFC$^\omega$ that combines all these theories: the union of ZFC, ZFC$^+$, ZFC^{++}, etc. This theory is still recursively axiomatizable, so the Second Incompleteness Theorem still applies: it can't prove its own consistency (if it is consistent). So we can keep adding consistency statements, creating ZFC$^{\omega+1}$, ZFC$^{\omega+2}$, ..., ZFC$^{\omega+\omega}$, and so on, up through the ordinals until we reach a point (the "Church-Kleene ordinal" ω_1^{CK}) where the union of the previous theories can no longer be captured in the language of ZFC.

Exercise 10.4 Formulas that satisfy P1–P3 are often called *provability predicates*. But this is misleading. Show that the formula $\text{SENT}(x)$ that represents (in

PA) the property of coding an $\mathfrak{L}_A$-sentence satisfies P1–P3.

Exercise 10.5 The Second Incompleteness Theorem applies to any predicate satisfying P1–P3: the predicate $\text{PROV}_T(x)$ that figures in the Theorem doesn't have to be defined as $\exists y\, \text{PRF}_T(y, x)$ from a predicate $\text{PRF}_T(x, y)$ that represents the proof relation of T. What do we learn if we apply the Second Incompleteness Theorem to $\text{SENT}(x)$?

Exercise 10.6 Let PA^+ be obtained from PA by adding $\text{PROV}_{\text{PA}}(\ulcorner \bot \urcorner)$ as a new axiom. Is this theory consistent? Is it ω-consistent? (Hint: remember that $\text{PROV}_{\text{PA}}(\ulcorner \bot \urcorner)$ abbreviates $\exists y\, \text{PRF}_{\text{PA}}(y, \ulcorner \bot \urcorner)$.)

Exercise 10.7 Using exercise 10.2, explain why PA can't prove the negation of $\neg\text{PROV}_{\text{PA}}(\ulcorner \bot \urcorner)$, unless it is inconsistent.

10.2 Löb's Theorem

For a suitable theory T, the Diagonal Lemma allows us to construct a Gödel sentence G that says of itself that it is unprovable in T. We can also construct a sentence H that says of itself that it is provable. More formally, if we apply the (Syntactic) Diagonal Lemma to the formula $\text{PROV}_T(x)$, we get a sentence H such that

$$\vdash_T H \leftrightarrow \text{PROV}_T(\ulcorner H \urcorner). \tag{D}$$

This is called the *Henkin sentence* for T. It's easy to show that no consistent theory can prove its Gödel sentence. For the Henkin sentence, the situation is less clear. Is H provable in T? This question was raised by Leon Henkin in 1952, and answered by Martin Löb in 1955. Löb showed that if T is consistent, recursively axiomatizable, and sufficiently strong, and it proves $\text{PROV}_T(\ulcorner A \urcorner) \to A$ for some sentence A, then it also proves that sentence A. This is known as *Löb's Theorem*. Since (D) gives us $\vdash_T \text{PROV}_T(\ulcorner H \urcorner) \to H$, the Theorem entails $\vdash_T H$. The Henkin sentence H is indeed provable in T.

The proof of Löb's Theorem resembles the following proof that Santa Claus exists.

Let S be the sentence 'if S is true then Santa Claus exists'. This being a conditional, we can try to prove it by deriving the consequent from the antecedent. So assume the antecedent: S is true. So if S is true then Santa Claus exists (for this is what S says). Still assuming that S is true, we can infer (by Modus Ponens) that Santa Claus exists. Now we have derived the consequent of S from its antecedent. So we've proved S: we've proved that if S is true then Santa Claus exists. And since we've proved S, we can infer by Modus Ponens that Santa Claus exists.

This line of reasoning is known as *Curry's Paradox*. It clearly works for any consequent A in place of 'Santa Claus exists'. In the following proof of Löb's Theorem, we replace 'is true' (which is not expressible in $\mathfrak{L}_A$) by 'is provable'. The assumption $\vdash_T \text{PROV}_T(\ulcorner A \urcorner) \to A$ of Löb's Theorem gives us one direction of the Tarski biconditional for A. In the presence of P1–P3, that's enough to run a Curry-style argument and show that $\vdash_T A$.

Theorem 10.2: (Löb's Theorem)

If T is a consistent, recursively axiomatizable theory in which diag is representable, and $\text{PROV}_T(x)$ satisfies P1–P3, then the following holds for any sentence A in the language of T:

$$\text{If } \vdash_T \text{PROV}_T(\ulcorner A \urcorner) \to A \text{ then } \vdash_T A.$$

Assume that $\vdash_T \text{PROV}_T(\ulcorner A \urcorner) \to A$. Applying the Syntactic Diagonal Lemma to the formula $\text{PROV}_T(x) \to A$, we get a sentence S such that $\vdash_T S \leftrightarrow (\text{PROV}_T(\ulcorner S \urcorner) \to A)$. Using the box notation, we now reason as follows.

1.	$\vdash_T S \leftrightarrow (\Box S \to A)$	(Diagonal Lemma)
2.	$\vdash_T S \to (\Box S \to A)$	(from 1)
3.	$\vdash_T \Box(S \to (\Box S \to A))$	(from 2 by P1)
4.	$\vdash_T \Box S \to \Box(\Box S \to A)$	(from 3 by P2)
5.	$\vdash_T \Box S \to (\Box\Box S \to \Box A)$	(from 4 by P2)
6.	$\vdash_T \Box S \to \Box\Box S$	(P3)
7.	$\vdash_T \Box S \to \Box A$	(from 5 and 6)
8.	$\vdash_T \Box A \to A$	(assumption)
9.	$\vdash_T \Box S \to A$	(from 7 and 8)
10.	$\vdash_T S$	(from 1 and 9)
11.	$\vdash_T \Box S$	(from 10 by P1)

12. $\vdash_T A$ (from 9 and 11) □

In line 1 of this proof, the Diagonal Lemma is used to construct the sentence S that "says that" (is provably equivalent to) 'if S is provable then A'. As in Curry's Paradox, T can derive that if S is provable then A (line 9). From this, T infers that S is true (line 10), and thereby that S provable (line 11), from which it infers A by Modus Ponens (line 12).

Exercise 10.8 Prove the converse of Löb's Theorem: if $\vdash_T A$, then $\vdash_T \text{PROV}_T(\ulcorner A \urcorner) \rightarrow A$.

From Löb's Theorem, it is a short step to the Second Incompleteness Theorem. Assume T can prove $\neg\text{PROV}_T(\ulcorner \bot \urcorner)$. By propositional logic, this entails $\text{PROV}_T(\ulcorner \bot \urcorner) \rightarrow \bot$. By Löb's Theorem, it follows that T can prove $\bot$. So if T can prove its own consistency, then T is inconsistent.

Löb's Theorem also entails a version of Tarski's Theorem on the undefinability of truth. Recall that a predicate $W(x)$ is a *truth predicate* for a theory T if T can prove all the Tarski biconditionals

$$W(\ulcorner A \urcorner) \leftrightarrow A.$$

Suppose that $W(x)$ is a truth predicate for T. Then $W(x)$ is also a provability predicate for T. Assume that T is a recursively axiomatizable theory in which diag is representable. Since $\vdash_T W(\ulcorner A \urcorner) \rightarrow A$ for all A, it follows by Löb's Theorem that $\vdash_T A$ for all A. So T is inconsistent.

Exercise 10.9 Explain why a truth predicate for T is also a provability predicate for T.

Löb's Theorem highlights the difference between the formal concept of provability and the concept of truth. One might have expected that a sufficiently powerful theory would "know that" whatever it can prove is the case. That is, one might have expected that PROV_T should satisfy the Reflection principle from exercise 10.3:

Ref $\vdash_T \text{PROV}_T(\ulcorner A \urcorner) \rightarrow A$.

(A truth predicate would satisfy both Ref and its converse.) Löb's Theorem shows that if T is sufficiently strong and consistent then PROV_T satisfies only those instances of Ref for which A is already provable in T, in which case $\text{PROV}_T(\ulcorner A \urcorner) \rightarrow A$ follows by propositional logic.

Exercise 10.10 What's the difference between the hypothesis $\vdash_{\text{PA}} A$ and the arithmetical hypothesis $\text{PROV}_{\text{PA}}(A)$? Can one be true without the other?

Exercise 10.11 Return to the theory PA$^+$ axiomatized by adding $\text{PROV}_{\text{PA}}(\ulcorner \bot \urcorner)$ to the axioms of PA. (a) Explain why $\vdash_{\text{PA}^+} \text{PROV}_{\text{PA}^+}(\ulcorner \bot \urcorner)$. (b) Show that there is a sentence A for which $\vdash_{\text{PA}^+} \neg(\text{PROV}_{\text{PA}^+}(\ulcorner A \urcorner) \to A)$.

10.3 The logic of provability

If you are familiar with modal logic, the provability conditions P1–P3 will look familiar, especially when written with the box operator $\Box$. If you aren't familiar with modal logic, let me quickly introduce some background.

Propositional modal logic is an extension of classical propositional logic. The language $\mathfrak{L}_M$ of propositional modal logic is obtained from $\mathfrak{L}_0$ by adding the unary sentence operator '$\Box$'. Thus whenever A is an $\mathfrak{L}_M$-sentence, then so is $\Box A$. The intended meaning of the box varies from application to application; often it is used to express some kind of necessity.

Hilbert-style proof systems for propositional modal logic extend the propositional calculus by axioms and inference rules that govern the behaviour of the box. A well-known proof system adds one axiom schema and one rule of inference:

K	$\Box(A \to B) \to (\Box A \to \Box B)$
Nec	From A one may infer $\Box A$

This system is known as K. (So 'K' names both a proof system and an axiom schema. The letter stands for Saul Kripke.) Stronger systems can be obtained by adding further axioms. For example, the system K4 adds an axiom schema known (for obscure historical reasons) as '4':

4 $\quad \Box A \to \Box\Box A$;

the system S4 adds both 4 and Ref (more commonly known as 'T'):

Ref $\quad \Box A \to A$.

Any system that can be obtained by adding axiom schemas to K is called a *normal modal logic*.

Above I used the "box notation" to simplify sentences of $\mathfrak{L}_A$. Sentences in the "box notation" look just like sentences of $\mathfrak{L}_M$. To make this more explicit, let t be a function that maps each sentence letter p of $\mathfrak{L}_M$ to an $\mathfrak{L}_A$-sentence $t(p)$. We can extend any such t to a mapping from $\mathfrak{L}_M$-sentences to $\mathfrak{L}_A$-formulas, as follows:

$$
\begin{aligned}
\widetilde{p}^{\,t} &= t(p)\\
\widetilde{\neg A}^{\,t} &= \neg\widetilde{A}^{\,t}\\
\widetilde{A \to B}^{\,t} &= \widetilde{A}^{\,t} \to \widetilde{B}^{\,t}\\
\widetilde{\Box A}^{\,t} &= \text{PROV}_T(\ulcorner \widetilde{A}^{\,t} \urcorner)
\end{aligned}
$$

Here I use '$\widetilde{A}^{\,t}$' to denote the output of the mapping for input A. For example, assume that $t(p)$ is '$0=0$'. Then $\widetilde{p}^{\,t}$ is '$0=0'$, $\widetilde{\neg p}^{\,t}$ is '$\neg(0=0)$', and $\widetilde{\Box p}^{\,t}$ is '$\text{PROV}_T(\ulcorner 0=0 \urcorner)$'.

Now assume that $\text{PROV}_T(x)$ satisfies P1–P3. In "box notation", P1, P2, and P3 are Nec, K, and 4, respectively. We might therefore expect that whenever an $\mathfrak{L}_M$-sentence A is provable in the modal system K4, then $\widetilde{A}^{\,t}$ is provable in T. We'll confirm this below.

The full modal logic of provability isn't K4, however. It has another axiom schema, known as GL (for Gödel-Löb):

GL $\quad \Box(\Box A \to A) \to \Box A.$

The system of modal logic obtained by adding GL to K4 is also called GL. (It turns out that the 4 schema is redundant: it can be derived from GL.)

GL is the modal translation of a formalized version of Löb's Theorem. Löb's Theorem states that

If $\vdash_T \text{PROV}_T(\ulcorner A \urcorner) \to A$ then $\vdash_T A$.

The formalized version of Löb's Theorem lifts this into T:

$\vdash_T \text{PROV}_T(\ulcorner \text{PROV}_T(\ulcorner A \urcorner) \to A \urcorner) \to \text{PROV}_T(\ulcorner A \urcorner).$

It is easily derivable from Löb's Theorem itself:

Lemma 10.1

If T is a consistent, recursively axiomatizable theory in which diag is representable, and $\text{PROV}_T(x)$ satisfies P1–P3, then the following holds for any sentence A in the language of T:

$$\vdash_T \text{PROV}_T(\ulcorner \text{PROV}_T(\ulcorner A \urcorner) \to A \urcorner) \to \text{PROV}_T(\ulcorner A \urcorner).$$

Proof. Using the box notation, we need to show that T proves $\Box(\Box A \to A) \to \Box A$. Call this sentence S. We show $\vdash_T S$ by showing $\vdash_T \Box S \to S$, from which $\vdash_T S$ follows by Löb's Theorem.

1. $\vdash_T \Box(\Box(\Box A \to A) \to \Box A) \to (\Box\Box(\Box A \to A) \to \Box\Box A)$ (P2)
2. $\vdash_T \Box(\Box A \to A) \to \Box\Box(\Box A \to A)$ (P3)
3. $\vdash_T \Box(\Box A \to A) \to (\Box\Box A \to \Box A)$ (P2)
4. $\vdash_T \Box(\Box(\Box A \to A) \to \Box A) \to (\Box(\Box A \to A) \to \Box A)$ (1–3)
5. $\vdash_T \Box(\Box A \to A) \to \Box A$ (4, Löb's Thm.) □

The following theorem confirms that if $\text{PROV}_T(x)$ satisfies P1–P3, and an $\mathfrak{L}_M$-sentence A is provable in the modal logic GL, then $\widetilde{A}^t$ is provable in T.

Theorem 10.3: The Arithmetical Soundness Theorem

If $\text{PROV}_T(x)$ satisfies P1–P3, then for any $\mathfrak{L}_M$-sentence A, if $\vdash_{\text{GL}} A$ then $\vdash_T \widetilde{A}^t$.

Proof. Assume $\vdash_{\text{GL}} A$. This means that there is a sequence $A_1, A_2, \ldots, A_n$ of $\mathfrak{L}_M$-sentences such that A_n is A and each A_k in the sequence is either an axiom of GL, an instance of the classical propositional axioms A1–A3, or follows from previous sentences by MP or Nec. We show by induction on k that $\vdash_T \widetilde{A_k}^t$.

If A_k is an instance of A1-A3 then $\widetilde{A_k}^t$ is also an instance of A1-A3, and so $\vdash_T \widetilde{A_k}^t$.

If A_k is an instance of K then $\vdash_T \widetilde{A_k}^t$ by P2,

If A_k is an instance of 4 then $\vdash_T \widetilde{A_k}^t$ by P3.

If A_k is an instance of GL, then $\vdash_T \widetilde{A_k}^t$ by Lemma 10.1.

If A_k follows by MP from A_i and A_j with $i, j < k$, then $\widetilde{A_k}^t$ follows by MP from $\widetilde{A_i}^t$

and $\widetilde{A_j}^t$. By induction hypothesis, $\vdash_T \widetilde{A_i}^t$ and $\vdash_T \widetilde{A_j}^t$. So $\vdash_T \widetilde{A_k}^t$.

Assume A_k follows by Nec from A_i with $i < k$. Then $\widetilde{A_k}^t$ is $\text{PROV}_T(\ulcorner \widetilde{A_i}^t \urcorner)$. By induction hypothesis, $\vdash_T \widetilde{A_i}^t$. So $\vdash_T \widetilde{A_k}^t$ by P1. □

The converse of Theorem 10.3 also holds: whenever $\vdash_T \widetilde{A}^t$, then $\vdash_{\text{GL}} A$. This *arithmetical completeness theorem* was proved by Robert Solovay in 1976. The proof is too hard to get into here.

Theorem 10.3 shows that we can use propositional modal logic to establish results about provability in theories like PA. Solovay's theorem shows that *all* general facts about provability in such theories can be established in this way.

For a simple illustration of how this works, here is a GL-proof of a formalized version of the Second Incompleteness Theorem, showing that if T meets the conditions for the Theorem, then it can prove that if it can prove its own consistency, then it is inconsistent.

1.	$\vdash_{GL} \neg\Box\bot \to (\Box\bot \to \bot)$	(propositional logic)
2.	$\vdash_{GL} \Box(\neg\Box\bot \to (\Box\bot \to \bot))$	(from 1 by Nec)
3.	$\vdash_{GL} \Box\neg\Box\bot \to \Box(\Box\bot \to \bot)$	(from 2 by K)
4.	$\vdash_{GL} \Box(\Box\bot \to \bot) \to \Box\bot$	(GL)
5.	$\vdash_{GL} \Box\neg\Box\bot \to \Box\bot$	(from 3 and 4)

By Theorem 10.3, it follows that $\vdash_T \text{PROV}_T(\ulcorner \neg\text{PROV}_T(\ulcorner \bot \urcorner) \urcorner) \to \text{PROV}_T(\ulcorner \bot \urcorner)$.

For another illustration, one can show that (for any sentence A)

$$\vdash_{GL} \Box(A \leftrightarrow \Box\neg A) \to \Box(A \leftrightarrow \Box\bot).$$

Among other things, this tells us what theories like PA will make of a sentence N that "says of itself" that its negation is provable. We get such a sentence by applying the Diagonal Lemma to the formula $\text{PROV}_T(\ulcorner \neg x \urcorner)$:

$$\vdash_T N \leftrightarrow \text{PROV}_T(\ulcorner \neg N \urcorner).$$

By P1, this entails

$$\vdash_T \text{PROV}_T(\ulcorner N \leftrightarrow \text{PROV}_T(\ulcorner \neg N \urcorner) \urcorner).$$

The above result from GL gives us

$$\vdash_T \text{PROV}_T(\ulcorner N \leftrightarrow \text{PROV}_T(\ulcorner \neg N \urcorner) \urcorner) \to \text{PROV}_T(\ulcorner N \leftrightarrow \text{PROV}_T(\ulcorner \bot \urcorner) \urcorner).$$

So by Modus Ponens,

$$\vdash_T \text{PROV}_T(\ulcorner N \leftrightarrow \text{PROV}_T(\ulcorner \bot \urcorner)\urcorner).$$

If T is sound, this entails

$$\vdash_T N \leftrightarrow \text{PROV}_T(\ulcorner \bot \urcorner).$$

So the sentence N that says that its negation is provable in PA is equivalent in PA to the statement that PA is inconsistent. Since PA can neither prove nor disprove its own consistency (assuming it is consistent), it can neither prove nor disprove N.

Exercise 10.12 Show that $\vdash_{GL} \Box\neg\Box A \to \Box A$.

Exercise 10.13 Let GL+Ref be the system obtained by adding the axiom schema Ref to GL. Show that $\Box\bot$ is provable in GL+Ref. Is GL+Ref consistent?

Exercise 10.14 Show that $\Box\bot$ is provable in the system GL+5 obtained by adding the axiom schema 5 to GL:

5 $\neg\Box A \to \Box\neg\Box A$.

Exercise 10.15 Show that $\vdash_{GL} (G \leftrightarrow \neg\Box G) \to (G \leftrightarrow \neg\Box\bot)$. (What does this tell us about the Gödel sentence G?)

Exercise 10.16 How should we restrict the Nec rule if we wanted to allow for deductions from premises in our calculus for normal modal logics?

10.4 Chaitin's Incompleteness Theorem

In 1974, Gregory Chaitin proved a version of the First Incompleteness Theorem that connects it to issues of computability and opens a different route to the Second Incompleteness Theorem.

Recall that a Turing machine takes as input a finite pattern of strokes and blanks on its tape (ignoring the blanks to left of the first stroke and to the right of the last stroke). Its output, if it halts, is also a finite pattern of strokes and blanks. Every such pattern can be produced by a Turing machine. In fact, for every finite pattern of strokes and blanks, there are infinitely many Turing machines that produce it, starting with a blank tape. We might be interested in the most efficient way to generate a given pattern: what is the shortest Turing machine program that produces it?

The *program* of a Turing machine is a string that lists all its instructions – all quintuples like $\langle q_0, B, 1, R, q_1 \rangle$ that specify what machine does if it scans a certain symbol in a certain state. We've seen in Section 6.3 that these instructions can be coded as patterns of strokes and blanks and fed to a universal Turing machine, which will execute the program.

Let's define the *complexity* of a pattern as the length of the shortest program that produces it. (This is somewhat imprecise, but the imprecision will do us no harm. The more precise concept is called *Kolmogorov complexity*.)

Some patterns can be produced by very short programs, others require very long programs. As a rule, short and regular patterns can be produced by short programs, while long and disorderly patterns tend to require long programs: they have high complexity.

It's easy to see that there is no limit to how complex a pattern can be. For any number n, only finitely many patterns can be produced by programs of length up to n; all other patterns require longer programs: they have greater complexity.

You'd think that for any number n, we can easily find examples of patterns whose complexity is demonstrably greater than n. Chaitin's Incompleteness Theorem shows that this is not so. In essence, it says that for any recursively axiomatized and consistent theory that knows some basic facts about Turing machines, there is a particular number L such that the theory can't prove of any pattern that its complexity is greater than L.

To state this more precisely, consider the relation H that holds between a Turing machine program M, a pattern of strokes and blanks S, and a number t if M halts with output S after t steps. This relation is decidable. (With a concrete coding of programs and patterns, it can easily be shown to be recursive.) So it is representable in any sufficiently strong arithmetical theory – e.g., in any extension of Q – by some formula $\mathrm{H}(x, y, z)$. Let $\mathrm{LEN}(x, y)$ represent the (obviously computable) function that maps each program to its length. We can now express the claim that the complexity of a pattern S exceeds a number n as the claim that there is no program x of length $\leq n$ that produces S after some number of steps t: $\forall x \forall y \forall t\, (\mathrm{LEN}(x, y) \land y \leq \overline{n} \to \neg \mathrm{H}(x, \ulcorner S \urcorner, t))$. Abbreviate this as $\mathrm{COMPEXC}(S, n)$.

Theorem 10.4: Chaitin's Incompleteness Theorem

If T is a consistent, recursively axiomatizable extension of Q, there is a number L such that for any pattern S, T cannot prove that the complexity of S exceeds L: $\nvdash_T \text{COMPEXC}(S, L)$.

Proof sketch. Since T is recursively axiomatizable, we can design a Turing machine M_T that takes a number k as input and searches through all proofs in T until it finds a proof of a sentence of the form $\text{COMPEXC}(S, \overline{k})$; it then outputs S and halts.

We know from section 6.2 that there's a (fairly simple) machine that doubles the number of strokes on the tape. For any number $n > 0$, we can therefore design another machine M_n that (when started on a blank tape)

1. writes a single stroke, then
2. doubles the number of strokes on the tape n times, then
3. moves to the left end of all the strokes, then
4. calls M_T.

This machine writes 2^n strokes on the tape and then calls M_T. If there is a proof in T that the complexity of some pattern S is greater than 2^n, M_n will find some such proof; it will then output S and halt. If there is no such proof, M_n runs forever.

The program for M_n has length $d \cdot n + c$, where d is the length of the doubling machine's program and c is a constant for the length of M_T's program plus whatever is needed for writing the initial stroke and for moving to the left end of a block of strokes.

Since $d \cdot n + c$ grows more slowly than 2^n, there must be a number k such that

$$d \cdot k + c < 2^k.$$

Now suppose for reductio that there is a pattern S for which T can prove that its complexity exceeds $L = 2^k$. Then M_k will output some such S and halt. Since the program for M_k has length $d \cdot k + c$ and outputs S, the complexity of S is at most $d \cdot k + c$, which is less than L.

Since H represents the halting-with-bound relation, and M_k produces S after some number of steps t, T proves $\text{H}(\ulcorner M_k \urcorner, \ulcorner S \urcorner, \overline{t})$. T also knows the length of M_k's program: it can prove $\text{LEN}(\ulcorner M_k \urcorner) = \overline{d \cdot k + c}$. So T proves that there is a program of length $d \cdot k + c$ that produces S within t steps. By assumption, however, T also proves

COMPEXC$(S, \overline{2^k})$, which states that there is no program of length $\leq 2^k$ that produces S after any number of steps. So T is inconsistent. □

A nice aspect of this proof is that it works for any reasonable definition of complexity. (That's why we didn't need to be very precise about this.) It also works for programs in ordinary programming languages like Python or JavaScript, rather than Turing machine programs, and for data structures that aren't just patterns of strokes and blanks. Among other things, it shows that there is a number L such that we can't prove (in, say, ZFC) of any specific string of bits that it requires a Python program longer than L to be produced.

We can estimate the value of L. One would certainly need a lot of instructions to program the machine M_T (in the proof of Theorem 10.4) that takes a number k as input and then searches through all proofs in T until it finds a proof of some statement of the form 'the complexity of S is greater than k'. But one wouldn't need an astronomical number of instructions. 10,000 should be enough. If the doubling machine needs 10 instructions, M_n needs around $10n+10,000$ instructions, each of which has a fixed length. If the average length of these instructions is, say, 10 (never mind how this is measured), $d \cdot k + c$ will be around $100,000 + 100k$. The smallest k with $100,000 + 100k < 2^k$ is $k = 17$. This puts L at $2^{17} = 131,072$. For less cumbersome kinds of programs, L is even smaller. For Python and bit strings, it is well under 10,000.

That's an astonishingly small number. Imagine a 3D movie of the entire observable universe for its first billion years, at atomic resolution. Could you write a Python program that generates this movie, in under 10 KB, without using external resources? Surely not! Chaitin's Incompleteness Theorem implies that while this may be true, it can't be proved.

Exercise 10.17 Show that the complexity function that assigns to any finite pattern of strokes and blanks its complexity is not computable.
(Hint: Assume some machine M computes the complexity function. For any number n, one can then design a machine M_n that goes through all finite patterns until it finds one whose complexity exceeds n, then outputs that string and halts. The program for M_n is not much longer than that of M. Now derive a contradiction.)

Like Gödel's First Incompleteness Theorem, Chaitin's Incompleteness Theorem shows that certain arithmetical truths are unprovable in PA or ZFC (assuming these theories are sound). In 2010, Shira Kritchman and Ran Raz pointed out that there is also a route from Chaitin's Theorem to the Second Incompleteness Theorem.

We begin with an apparent paradox.

There is a finite number of programs of length $\leq L$. Let N be that number plus 1.

Each number n between 1 and N can be coded as a pattern of $n + 1$ strokes. Since there are more such numbers than programs of length $\leq L$, at least one of them must require a program longer than L to be printed. How many, exactly, require a program longer than L?

Suppose there is exactly one number x between 1 and N that requires a program longer than L to be printed. In that case, we can find it. We can run all Turing machines whose program has length $\leq L$. By assumption, at some point, all the $N - 1$ numbers that can be printed by such machines have been printed. We know that there is at least one number between 1 and N that requires a longer program. So we know that the remaining number must be the one that requires a longer program. At this point, our procedure proves that the complexity of that remaining number is greater than L. But by Chaitin's Incompleteness Theorem, we can't prove this!

This argument seems to show that there can't be a single number between 1 and N that requires a program longer than L to be printed. Well, suppose there are two. In that case, we can find them. We run through all Turing machines with programs of length $\leq L$ until all the $N - 2$ numbers that can be printed by such machines have been printed. We've just shown that there must be at least two numbers between 1 and N that require a program longer than L to be printed. So we can identify these two numbers as the ones that haven't been printed yet. We have therefore proved that the complexity of these two numbers is greater than L. By Chaitin's Incompleteness Theorem, this is impossible.

So there can't be exactly two numbers between 1 and N that require a program longer than L to be printed. Suppose there are three…

Continuing this line of thought up to N, we can seemingly show that every number between 1 and N is too complex to be printed with a program of length $\leq L$. But this is evidently false. We can easily show that, say, the number 1 can be printed with a program much shorter than L.

Kritchman and Raz show where the paradoxical argument fails if it is spelled out formally. This requires a theory in which one can formalize the proof of Chaitin's Theorem. Q is too weak for this, but PA is strong enough. More specifically, PA can prove that there is a number L such that PA can't prove of any number x that the complexity of x is greater than L – unless PA is inconsistent, in which case, of course, it proves everything. PA can also prove that there is at least one number between 1 and N whose complexity is greater than L. And it can prove that if there is exactly one such number then PA can find it, by showing that all the other numbers $y < N$ have complexity at most L, and inferring that the remaining number has complexity greater than L. So PA can prove that if there is exactly one number between 1 and N then PA can prove that the complexity of that number is greater than L.

In the argument above, I said that this contradicts Chaitin's Incompleteness Theorem. But Chaitin's Theorem for PA doesn't quite say that PA can't prove that the complexity of x is greater than L. It says that PA can't prove this *if PA is consistent*.

If PA could prove its own consistency, it could use Chaitin's Theorem to conclude that there can't be exactly one number between 1 and N with complexity greater than L. It could similarly show that there can't be exactly two such numbers, and so on. It would reach the conclusion that every number between 1 and N has complexity greater than L, which it can also refute, as it can prove that 1 has complexity less than L. Thus, if PA could prove its own consistency, it would be inconsistent.

10.5 Philosophy of Mind

Gödel's Incompleteness Theorems, and their corollaries, have profound implications for the foundations of mathematics. They derailed Hilbert's program. They draw a wedge between truth and provability, showing that there is no way to capture all mathematical truths in an axiomatic system. Some have argued that Gödel's theorems also have profound implications for our understanding of the human mind: they show that our mathematical capacity goes beyond what any mechanical system (any computer) can achieve.

In its basic form, this argument goes as follows.

Let S be the set of mathematical sentences that I accept as true. S includes the axioms of PA. Let S^+ be the set of sentences entailed by S. If my mind is equivalent to a Turing machine, S is computably enumerable, and S^+ is a computably axiomatizable extension of PA. I can then go through the proof of Gödel's First Incompleteness Theorem to construct a sentence G that is true, but not in S^+. This is impossible: since I realize that G is true, G is part of my mathematical knowledge: it must be in S^+! Hence my mind is not equivalent to a Turing machine.

An initial problem with this argument is that the inference to the truth of G requires the assumption that S^+ is consistent. Gödel's proof shows that if S^+ is axiomatizable *and consistent*, then G is not in S^+. To reach the conclusion that G is true, the argument therefore assumes that the set S^+ includes the statement that S^+ is consistent. Obviously, S^+ is consistent iff S is consistent. So the argument assumes that the set S of my mathematical beliefs includes the statement that this very set is consistent. It's not enough to believe that "my mathematical beliefs are consistent"; I would also have to be aware of the fact that my mathematical beliefs comprise precisely the set S. Thus one way to escape the argument is to hold that we cannot be fully aware of our mathematical beliefs.

There are other problems with the argument. The concepts of mathematical knowledge and belief are surprisingly difficult to model consistently, especially if we want to allow for beliefs about one's own beliefs.

Suppose we extend the language of arithmetic by a predicate B for belief. On its intended interpretation, $B(\ulcorner A \urcorner)$ is meant to express that I accept the sentence A. (We could use special quote marks to refer to sentences, rather than Gödel numbers. This would make no difference.) We could also add a predicate K for knowledge.

We might now want to lay down some axioms for B and K. For example, since knowledge is factive (what is known is true), a minimal theory of mathematical knowledge and belief should include the schema Ref_K:

$$\text{Ref}_K \quad \vdash_T K(\ulcorner A \urcorner) \to A.$$

Suppose such a theory also includes the axioms of Q. In fact, let's simply consider the theory T that adds Ref_K to Q. By the Diagonal Lemma, there is a sentence G such that

$$\vdash_T G \leftrightarrow \neg K(\ulcorner G \urcorner). \tag{D}$$

By Ref_K, $\vdash_T K(\ulcorner G \urcorner) \to G$. Combining this with (D), we get $\vdash_T K(\ulcorner G \urcorner) \to \neg K(\ulcorner G \urcorner)$, which entails $\vdash_T \neg K(\ulcorner G \urcorner)$. By (D) again, we get $\vdash_T G$.

This shows that the sentence G that figures in (D) is derivable from the axioms of Q and an instance of Ref_K. One would think that I could know the axioms of Q and the relevant instance of Ref_K. Going through the above reasoning, I could thereby come to know G. But this is incompatible with T, for we've just seen that $\vdash_T \neg K(\ulcorner G \urcorner)$!

This puzzle is known as the *Knower Paradox*. If we hold fixed classical logic and the factivity of knowledge (as expressed by Ref_K), the only way to avoid contradiction is to deny that I can come to know G by competently going through its proof. Notice that the above argument against the equivalence between my mind and a Turing machine involved a very similar assumption: that I could come to know the Gödel sentence G of S^+ by going through the proof of Gödel's Theorem.

The situation for belief is arguably even worse. If we add B to $\mathfrak{L}_A$, we can construct a formula $\text{BC}(x, y)$ saying that there is a code c of a sequence of sentences $A_1, \ldots, A_n$ such that $B(\text{ENTRY}(c, i))$ for each $1 \leq i \leq n$, and $\exists y \text{PRF}_{\text{PA}}(y, \ulcorner A_1 \wedge \ldots \wedge A_n \to A \urcorner)$. That is, $\text{BC}(\ulcorner A \urcorner)$ is true iff A is provable in PA from premises that I believe.

Now assume that the set of sentences that I believe is decidable and consistent with PA. (This is surely possible.) Let B^* be the set of sentences that are provable in PA from premises that I believe. B^* is a consistent, axiomatizable extension of PA. Without further assumptions about B, one can show that $\textsc{bc}(x)$ satisfies P1–P3. So Löb's Theorem applies: whenever B^* contains $\textsc{bc}(\ulcorner A \urcorner) \rightarrow A$, it also contains $\textsc{bc}(\ulcorner A \urcorner)$.

This is highly counterintuitive. For example, one might think that I could believe that $1{+}1 = 3$ does not follow in PA from my mathematical beliefs. (It certainly seems to me as if I believe this!) But if $\neg\textsc{bc}(\ulcorner 1+1=3 \urcorner)$ is in B^* then so is its tautological consequence $\textsc{bc}(\ulcorner 1+1 = 3 \urcorner) \rightarrow 1+1 = 3$. By Löb's Theorem, it follows that '$1+1 = 3$' is in B^*. That is, I can only believe that '$1+1 = 3$' does not follow from my mathematical beliefs if it actually follows from my mathematical beliefs!