

# FORMAL BACKGROUND FOR THEORIES OF TRUTH

RICHARD G. HECK, JR.

Tarski proved two important results about truth.<sup>1</sup> First, he provided a method that, applied to any of a wide class of languages, yields a “materially adequate” theory of truth for that language, the ‘object language’: The theory entails all instances of schema T for the object language, and this theory is both consistent and mathematically useful. Second, Tarski showed that the language in which the theory of truth is given—the ‘meta-language’—must always be a proper extension of the language the theory is about—the ‘object language’. That is: For *no* language of a certain minimum expressive power is it possible to give a consistent theory of truth for that very language in that very language. In this note, we will prove this latter result. A separate note discusses the former.<sup>2</sup>

## 1. LANGUAGES AND THEORIES

We need first carefully to distinguish between a *language* and a *theory*.<sup>3</sup> A language is just a collection of symbols. Any collection of symbols may be regarded as a language, but we shall be interested here only in *first-order* languages: Such languages are generated, in the obvious sense, from basic logical symbols and variables, plus as many ‘non-logical’ symbols as one likes. These may include constants, such as ‘0’, function symbols, such as ‘+’ or ‘g’, and predicate letters, such as ‘>’ or ‘F’, of various numbers of arguments. (A sentence-letter is a zero-place predicate-letter.) We assume that these languages have the usual syntax of first-order logic. We regard languages as *fixed* and identify them with their stock of non-logical expressions.

---

<sup>1</sup>The purpose of this note is to make it possible for those have a firm grasp of basic logic to acquire a solid understanding of Tarski’s Theorem. One can acquire such an understanding, I believe, without wading through all the messy details that—as conceptually interesting as they may be in their own right—throw no real light on Tarski’s Theorem itself. (I have in mind, for example, the Beta Function Lemma.) Of course, the only way to acquire an understanding of Tarski’s Theorem is to work one’s way through a proof of it. But it would, it seems to me, be a shame if students who are, for whatever reason, unwilling or unable to take a Gödel’s Theorem-type logic course were, for that same reason, barred from any real understanding of this important result. Hence, this note. Of course, the proof of my pedagogical orientation—my decisions regarding what casts light on the theorem and what does not—will be in the proving of the theorem itself.

The proof of Tarski’s Theorem offered below draws heavily upon the presentation by Boolos and Jeffrey (1989). See especially Ch. 15. More generally, my own understanding of the result was heavily influenced by discussions with George and by his general logico-philosophical orientation.

For an illuminating discussion of Gödel numbering, syntax and the like, see Boolos’s *The Logic of Provability* (1993).

<sup>2</sup>See my “Tarski’s Theory of Truth”, available on my web site.

<sup>3</sup>It’s not clear how Tarski himself uses these terms in “The Concept of Truth”. In many of his uses of the term, a ‘language’ seems to be what we are calling a theory (which carries with it a language). Whether this is just a difference of terminology or whether Tarski doesn’t clearly distinguish between language and theory is what is not so clear.

For our purposes, the most important language is the *language of arithmetic*, whose (atomic) non-logical symbols are the constant ‘0’, the one-place function-symbol ‘S’, two two-place function-symbols ‘+’, and ‘×’, and a two-place relation symbol ‘<’.<sup>4</sup> (Another important language is the language of set theory, whose only non-logical symbol is the two-place relation symbol ‘∈’.) So ‘0 + Sx’ is an (open) term of this language and ‘ $\forall x \exists y (x = y + Sy)$ ’ is a sentence. On the other hand, neither ‘ $\exists z (+x)$ ’ nor ‘ $\forall x (Fx \rightarrow Gx)$ ’ is a sentence of the language of arithmetic. The former is not well-formed and so it is not a sentence of *any* (first-order) language. The latter is a sentence of some languages—of any language that contains both ‘F’ and ‘G’. But it is not a sentence of the language of arithmetic, which does not contain either.

As syntax is usually developed, formally, much of the previous paragraph is false. Strictly speaking, there are always supposed to be parentheses around the arguments of function-symbols and predicate-letters (thus: ‘(y) + (S(y))’, not ‘y + Sy’), and around conjuncts, disjuncts, etc., to guarantee unique readability. In practice, when nothing depends upon their visibility, we write parentheses with invisible ink. But it is worth recalling, from time to time, that, strictly speaking, the string ‘ $\forall x \exists y (x = y + Sy)$ ’ is *not* a sentence of the language of arithmetic, though the string ‘ $\forall x (\exists y ((x) = ((y) + (S(y))))))$ ’ is. That said, however, we shall henceforth promptly forget about this point.<sup>5</sup>

It simplifies the following for there to be only finitely many primitive symbols in the language of arithmetic. Of course, the logical symbols, and special arithmetical symbols, are finite in number, but there are infinitely many variables. We can, however, construct our infinitely many variables from a finite alphabet, as follows:  $x, x', x''$ , etc.; a variable is thus an ‘x’ followed by a (possibly empty) string of prime-symbols. We shall, however, frequently abbreviate ‘x’ as: y; ‘x’ as: z, and so forth, to enhance readability.

An important method of proof in syntax is *induction on the complexity of expressions*. This method is similar to ‘mathematical’ induction. In arithmetic, one can show that every number has a certain property  $P$  by showing, first, that 0 has  $P$  and, second, that, if a number  $n$  has  $P$ , then the next number,  $n + 1$ , has  $P$ . This works because every number is the result of adding one to 0 some finite number of times. But similarly: Every term in the language of arithmetic is the result of applying certain syntactic operations to the basic terms, ‘0’ and the variables. So suppose we knew that ‘0’ had  $P$  and that every variable had  $P$ ; and suppose we knew further that, whenever terms  $t$  and  $u$  have  $P$ , so do the terms: ‘ $St$ ’, ‘ $t + u$ ’, and ‘ $t \times u$ ’. Then it would follow that every term has  $P$ . Similarly, suppose we knew that all atomic formulae had  $P$ —the atomic formulae, in this case, are of the form ‘ $t = u$ ’ and that, whenever two formulae  $A$  and  $B$  have  $P$ , so do: ‘ $\neg A$ ’, ‘ $A \vee B$ ’, ‘ $A \wedge B$ ’, ‘ $A \rightarrow B$ ’, ‘ $\exists v A$ ’ and ‘ $\forall v A$ ’, for any variable  $v$ . Then, again, it would follow that all formulae have  $P$ .

The language of arithmetic is an *interpreted* language, a language whose formulae and sentences mean something. What they mean is determined by what is called the ‘standard interpretation’ of the language. The standard interpretation is an interpretation, in the usual sense: It has a domain, and it assigns values to constants and extensions to function-symbols. In this case, the domain is the set of all natural numbers; ‘0’ denotes zero; ‘S’ denotes successor, or plus one; ‘+’ denotes addition; ‘×’, multiplication; and ‘<’, the less-than relation. More precisely, the extension of ‘S’

<sup>4</sup>The last is often, even usually, omitted, but its inclusion is important in the study of weak fragments of arithmetic.

<sup>5</sup>Of course, since the conventions that allow the invisibility of parentheses define recursive functions between ‘strict’ and ‘sloppy’ formulae, we can, if we like, allow the conventions to be part of the formal syntax. The closest anyone has come to a natural such treatment is, I think, Boolos, in *The Logic of Provability*.

is the set:  $\{ \langle x, y \rangle : y = x + 1 \}$ ; that of ‘+’ is:  $\{ \langle x, y, z \rangle : x + y = z \}$ ; etc. So we may now regard a given sentence, say, ‘ $\forall x \exists y (x = y + Sy)$ ’, as meaning something, in this case that every number is the sum of some number and its successor (roughly, that every number is odd), which is false.

Expressions like ‘ $SS0$ ’—that is, strings of ‘ $S$ ’s, followed by a ‘ $0$ ’—are called *numerals*. Note that, in the standard interpretation, a string of  $n$  ‘ $S$ ’s followed by ‘ $0$ ’ denotes the number  $n$ . Thus, ‘ $SS0$ ’ denotes the number 2 in the standard interpretation, and so what ‘ $SS0 = S0$ ’ says, as it were, is that  $2 = 1$ . We write  $\mathbf{n}$  to mean: the numeral that denotes  $n$ , that is, ‘ $\underbrace{S \dots S}_n 0$ ’. So, we may now write things like:  $\Sigma$  proves ‘ $\mathbf{2} \neq \mathbf{1}$ ’, to mean that  $\Sigma$  proves ‘ $SS0 \neq S0$ ’.

Note the use of the funny quotes, which are called corners and which are strictly speaking necessary. If  $\Sigma$  is a theory in the language of arithmetic, then  $\Sigma$  does *not* prove ‘ $\mathbf{2} \neq \mathbf{1}$ ’, for the simple reason that the expression ‘ $\mathbf{2} \neq \mathbf{1}$ ’ is not a formula of the language of arithmetic—and that for the simple reason that ‘ $\mathbf{2}$ ’ is not a symbol of the language of arithmetic but is, rather, a *name* of such a symbol, namely, of ‘ $SS0$ ’. (Of course, strictly speaking, ‘ $\neq$ ’ is not in the language of arithmetic either but is an abbreviation. But let us not get *too* pedantic.) Corners are a solution to this problem, introduced by Quine. Exactly how they work is not easily explained, but one does get the hang of it fairly quickly. In much technical writing nowadays, the corners are just written as ordinary quotes and context is left to distinguish them. Quotes themselves are even written as blanks, sometimes. We shall indulge in this practice ourselves from time to time. But we shall try not to overdo it.

So much for languages. On to theories.

**Definition.** A *theory* is a set of formulae of some fixed language, called the language of the theory. A theory is *stated in* or *formulated in* a language.

We shall typically use capital Greek letters, such as  $\Sigma$ , to range over theories. Note that any set of formulae constitutes a theory, although some theories will be more interesting than others.

Here are some examples of theories.

- (1) The set  $\{ \forall x \forall y (Sx = Sy \rightarrow x = y), \neg \exists x (0 = Sx) \}$  is a theory stated in the language of arithmetic. It is also a theory in the language  $\{0, S\}$ , where ‘ $0$ ’ is a constant; ‘ $S$ ’, a one-place function-symbol. Strictly speaking, those are really two theories, and a theory is a pair  $\langle \mathcal{T}, \mathcal{L} \rangle$  of a set of formulae and a language.
- (2) The *theory of groups* is the set containing the formulae:  $\forall x \forall y \forall z ((x + y) + z = x + (y + z), \forall x (x + 0 = x), \forall x \exists y (x + y = 0 \wedge y + x = 0)$ . It is stated in the language whose non-logical symbols are ‘ $0$ ’ and ‘+’.
- (3) The empty set is always a theory, though we do need to specify, in each case, what the language of the theory is.

We assume here a notion of *derivation* provided by ordinary first-order logic. We say that  $A$  is derivable from the set of formulae  $\Gamma$  if there is a formal derivation of  $A$  whose premises are all in  $\Gamma$ . Note that not all the sentences in  $\Gamma$  need be used as premises—and of course they will not be if  $\Gamma$  is infinite.

**Definition.** A formula  $A$  is a *theorem* of the theory  $\Sigma$  if, and only if,  $A$  is derivable from  $\Sigma$ .

We write:  $\Sigma \vdash A$ , to mean that  $A$  is a theorem of  $\Sigma$ . One also sees such things as:  $A \vdash B$  and  $\Sigma, A \vdash B$ —meaning that  $B$  is derivable from  $A$ , or from  $\Sigma$  together with  $A$ —where strictly we

should write:  $\{A\} \vdash B$ , or:  $\Sigma \cup \{A\} \vdash B$ . We also read ‘ $\Sigma \vdash B$ ’ as ‘ $\Sigma$  proves  $B$ ’. One also sees the notation:  $\vdash_{\Sigma} B$ , meaning:  $\Sigma \vdash B$ .

*The notion of derivability is not the same as the notion of implication.* Whether a theory  $\Sigma$  *implies* a given formula  $A$  is a matter of whether  $A$  is true under every interpretation that makes every formula in  $\Sigma$  true. Whether  $\Sigma$  *proves*  $A$  is a matter of whether a formal derivation of a certain sort exists. The completeness theorem for first-order logic tells us that  $\Sigma$  proves  $A$  if, and only if,  $\Sigma$  implies  $A$ . But this is a deep theorem about first-order logic not a trivial fact of terminology. It is, indeed, a *very* deep theorem, one whose analogue for other sorts of logics does not always hold.

As noted,  $\Sigma$  implies  $B$  if, and only if,  $B$  is true in every interpretation in which all formulae in  $\Sigma$  are true. Interpretations in which all formulae in  $\Sigma$  are true are thus of special interest. We call such an interpretation a *model* of  $\Sigma$ . So  $\Sigma$  implies  $B$  if, and only if,  $B$  is true in every model of  $\Sigma$ . If there is a model of  $\Sigma$  in which  $B$  is not true,  $\Sigma$  does not imply  $B$ , so  $B$  is not derivable from  $\Sigma$  (by the soundness theorem), so  $B$  is not a theorem of  $\Sigma$ . Such a model is called a *counter-model* for  $B$ .

All we need to know for present purposes is that *there is* a sound and complete system of axioms and rules for first-order logic available for use in derivations. For then,  $\Sigma$  implies  $A$  if, and only if,  $\Sigma$  proves  $A$ , using *any* such system of axioms and rules. We shall appeal to this fact frequently, without any explicit notice, and without bothering to formalize the logic.

## 2. SOME IMPORTANT PROPERTIES OF THEORIES

**Definition.** We say that a theory  $\Sigma$  is *consistent* if for no formula  $A$  is  $\lceil A \wedge \neg A \rceil$  derivable from  $\Sigma$ . If  $\Sigma$  is not consistent, we say that it is *inconsistent*.

Note that we cannot state this definition as: ... iff ‘ $p \wedge \neg p$ ’ is derivable from  $\Sigma$ . There is no guarantee that ‘ $p \wedge \neg p$ ’ is in the language of  $\Sigma$ . It is not, for example, in the language of arithmetic.

**Proposition 2.1.**  $\Sigma$  is consistent if, and only if, not every formula is derivable from  $\Sigma$ .

*Proof.* Exercise. □

**Definition.** A theory  $\Sigma$  is *complete* if, and only if, for every formula  $A$  in the language of the theory, either  $\Sigma \vdash A$  or  $\Sigma \vdash \neg A$ . If  $\Sigma$  is not complete, we say it is *incomplete*.

Note that every inconsistent theory is complete, since *both*  $A$  and  $\neg A$  will be derivable from it, for any formula  $A$ . Inconsistent (classical)<sup>6</sup> theories are boring, so let us mean by a complete theory a complete *consistent* theory.

If  $\Sigma$  is incomplete, there is a formula  $A$  such that neither  $A$  nor  $\neg A$  is derivable from  $\Sigma$  (and so there are infinitely many such formulae:  $A \wedge A$ ,  $A \wedge A \wedge A$ , etc).

**Definition.** A *closed theory* is one that is ‘closed under derivability’:  $\Sigma$  is closed if, and only if, if  $\Sigma \vdash A$ , then  $A \in \Sigma$ ; that is, iff every formula (in the language of  $\Sigma$ ) that is deducible from  $\Sigma$  is already in  $\Sigma$ .

---

<sup>6</sup>There are non-classical logics—so-called *paraconsistent* logics—in which contradictions do not imply everything and so for which proposition 2.1 is not provable. Inconsistent theories in such contexts need not be boring.

A consistent theory that is both complete and closed is said to be *maximal consistent*, because, as the following shows, it is *maximal* with respect to consistency: No consistent theory in the same language properly contains it.

**Proposition 2.2.** *Let  $\Sigma$  be a complete, closed theory. Let  $\mathcal{T}$  be any theory in the same language that properly contains  $\Sigma$ . Then  $\mathcal{T}$  is inconsistent.*

*Proof.* Let  $\Sigma$  and  $\mathcal{T}$  be as in the statement of the proposition. Then there is some formula  $A$  in the common language of  $\Sigma$  and  $\mathcal{T}$  such that  $A \in \mathcal{T}$  but  $A \notin \Sigma$ . Now, if  $\Sigma \vdash A$ , then  $A \in \Sigma$ , since  $\Sigma$  is closed, so  $\Sigma \not\vdash A$ . Since  $\Sigma$  is complete, we must then have that  $\Sigma \vdash \neg A$ . But then, since  $\Sigma \subseteq \mathcal{T}$ ,  $\mathcal{T} \vdash \neg A$ , too. But obviously  $\mathcal{T} \vdash A$ , since  $A \in \mathcal{T}$ , and so  $\mathcal{T} \vdash A \wedge \neg A$ , and hence  $\mathcal{T}$  is inconsistent.  $\square$

The following important fact is one we shall not need here, but it is well worth knowing. Most proofs of the completeness theorem rely upon it in one way or another.

**Proposition 2.3** (Lindenbaum's Lemma). *Every consistent set of formulae can be expanded to a maximal consistent set of formulae.*

*Proof.* Exercise.  $\square$

**Definition.** Let  $\mathcal{T}$  and  $\mathcal{T}^*$  be theories in the languages  $\mathcal{L}$  and  $\mathcal{L}^*$ , respectively, and assume that  $\mathcal{L} \subseteq \mathcal{L}^*$ . Then we say that  $\mathcal{T}^*$  is a *conservative extension* of  $\mathcal{T}$  if every theorem of  $\mathcal{T}^*$  in  $\mathcal{L}$  is already a theorem of  $\mathcal{T}$ .

More intuitively, if  $\mathcal{L} \subsetneq \mathcal{L}^*$ , then of course  $\mathcal{T}^*$  will have theorems that are not theorems of  $\mathcal{T}$ —simply because there are sentences in the language of  $\mathcal{T}^*$  that are not sentences in the language of  $\mathcal{T}$ . If  $\mathcal{T}^*$  is a conservative extension of  $\mathcal{T}$ , however, then these are the only ‘new’ theorems of  $\mathcal{T}^*$ . Every sentence that *might* be a theorem of  $\mathcal{T}$ —that is, every sentence in the language of  $\mathcal{T}$ —that is a theorem of  $\mathcal{T}^*$  is already a theorem of  $\mathcal{T}$ .

The easiest way to show that one theory is a conservative extension of another is via a simple model-theoretic argument.

**Theorem 2.4.** Let  $\mathcal{T}$  and  $\mathcal{T}^*$  be theories in the languages  $\mathcal{L}$  and  $\mathcal{L}^*$ , respectively, and assume that  $\mathcal{L} \subseteq \mathcal{L}^*$ . Assume further that every model of  $\mathcal{T}$  can be expanded to a model of  $\mathcal{T}^*$ .<sup>7</sup> Then  $\mathcal{T}^*$  is a conservative extension of  $\mathcal{T}$ .

*Proof.* Suppose  $\mathcal{T}^*$  is not a conservative extension of  $\mathcal{T}$ . Then there is a sentence  $A \in \mathcal{L}$  that is a theorem of  $\mathcal{T}^*$  but not a theorem of  $\mathcal{T}$ . By the completeness theorem, then, there is a model  $\mathcal{M}$  of  $\mathcal{T}$  in which  $A$  is not true. But by the assumption of the theorem,  $\mathcal{M}$  can be expanded to a model  $\mathcal{M}^*$  of  $\mathcal{T}^*$ . Since  $A \in \mathcal{L}$ , however,  $A$  must still be true in  $\mathcal{M}^*$ .<sup>8</sup>  $\square$

<sup>7</sup>A model  $\mathcal{M}^*$  of  $\mathcal{T}^*$  is an expansion of a model  $\mathcal{M}$  of  $\mathcal{T}$  if the two models agree on what they assign to the primitives of  $\mathcal{L}$ .

<sup>8</sup>This should be intuitively clear—what the model assigns to primitive expressions not in  $A$  cannot affect the value  $A$  has in the model—but it can also be proven rigorously. The relevant fact is: If  $\mathcal{M}^*$  is an expansion of  $\mathcal{M}$ , and if  $A \in \mathcal{L}$ , then  $A$  is true in  $\mathcal{M}$  iff  $A$  is true in  $\mathcal{M}^*$ . The proof is by induction on the complexity of  $A$  and is left as an exercise.

### 3. ARITHMETICAL THEORIES

*Robinson Arithmetic*, or  $Q$ , is the theory with the following eight axioms:<sup>9</sup>

- (1)  $\forall x(Sx \neq 0)$
- (2)  $\forall x\forall y(Sx = Sy \rightarrow x = y)$
- (3)  $\forall x(x + 0 = x)$
- (4)  $\forall x\forall y(x + Sy = S(x + y))$
- (5)  $\forall x(x \times 0 = 0)$
- (6)  $\forall x\forall y(x \times Sy = (x \times y) + x)$
- (7)  $\forall x(x \neq 0 \rightarrow \exists y(x = Sy))$
- (8)  $x < y \equiv \exists z(y = Sz + x)$

Note that all axioms of  $Q$  are true in the standard interpretation of the language of arithmetic. Moreover, as its namesake, Raphael Robinson, noted,  $Q$  is distinguished by the simplicity and clear mathematical content of its axioms. It's hard to imagine disputing  $Q$ , unless one wanted to dispute the very coherence of arithmetical notions.

In some ways,  $Q$  is a very weak theory. It is easy to see that  $Q$  does not, for example, prove that addition is commutative: That is, ' $\forall x\forall y(x + y = y + x)$ ' is not a theorem of  $Q$ . Even ' $\forall x(0 + x = x)$ ' turns out not to be a theorem of  $Q$ . In another sense, however,  $Q$  is quite a powerful theory, as we shall see.

**Exercise 3.1.** Show that the mentioned formulae are not theorems of  $Q$ .

$Q$  has a nice property we shall need later: It proves all basic arithmetical equalities and inequalities. By this we mean, for example, that whenever  $n$  is not  $m$ ,  $Q$  proves ' $\ulcorner n \neq m \urcorner$ '. Note carefully what this means, namely, that  $Q$  proves things like:  $S0 \neq S0$ .

**Proposition 3.2.**  $Q$  proves all basic arithmetical equalities and inequalities. That is:

- (1) If  $n = m$ ,  $Q \vdash \mathbf{n} = \mathbf{m}$ , and if  $n \neq m$ ,  $Q \vdash \mathbf{n} \neq \mathbf{m}$ .
- (2) If  $n + m = k$ , then  $Q \vdash \mathbf{n} + \mathbf{m} = \mathbf{k}$ , and if  $n + m \neq k$ , then  $Q \vdash \mathbf{n} + \mathbf{m} \neq \mathbf{k}$ .
- (3) If  $n \times m = k$ , then  $Q \vdash \mathbf{n} \times \mathbf{m} = \mathbf{k}$ , and if  $n \times m \neq k$ , then  $Q \vdash \mathbf{n} \times \mathbf{m} \neq \mathbf{k}$ .
- (4) If  $m < n$ , then  $Q \vdash \mathbf{m} < \mathbf{n}$ .

*Proof.* We prove (1) and leave the remainder as exercises.

That  $Q$  proves all true numerical identities is obvious: These are just logical truths of the form:  $S\dots S0 = S\dots S0$ . For the case of non-identities, the proof is by induction on  $m$ . NOTE: This is induction in the *meta-language*, not in  $Q$ . It can't be in  $Q$ , since  $Q$  doesn't have induction!

For the basis, we must show that  $Q$  proves  $\mathbf{n} \neq 0$ , if  $n \neq 0$ . But in that case,  $\mathbf{n}$  is  $S\dots S0$ , and  $S\dots S0$  will follow from axiom Q1. For the induction step, we suppose that  $Q$  proves  $\mathbf{n} \neq \mathbf{m}$  when  $n \neq m$  and show  $Q$  proves  $\mathbf{n} \neq S\mathbf{m}$  when  $n \neq Sm$ . Note first that this certainly holds if  $n = 0$ , by the preceding. So suppose  $n \neq 0$  and  $n \neq m$ . Then  $\mathbf{n}$  is  $S\mathbf{k}$ , for some numeral  $\mathbf{k}$ . Now if  $k = m$ , then  $k \neq Sm$ , and  $Q$  proves  $\mathbf{k} \neq S\mathbf{m}$  already, by the induction hypothesis. So suppose  $k \neq m$ . Then again by the induction hypothesis,  $Q$  proves  $\mathbf{k} \neq \mathbf{m}$ . By Q2, however,  $Q$  proves  $S\mathbf{k} = S\mathbf{m} \rightarrow \mathbf{k} = \mathbf{m}$ . But then it proves  $S\mathbf{k} \neq S\mathbf{m}$ , i.e.,  $\mathbf{n} \neq S\mathbf{m}$  as well.  $\square$

<sup>9</sup>When ' $<$ ' is not included in the language, the eighth is taken to define it.

Note carefully here what  $Q$  is being said to prove. It follows from what has been said, for example, that, for each  $n$  and  $m$ ,  $Q$  proves  $\mathbf{n} + \mathbf{m} = \mathbf{m} + \mathbf{n}$ . So  $Q$  is being said to prove each of infinitely many theorems. But  $Q$  does *not* prove that addition is commutative: It does not prove  $\forall x \forall y (x + y = y + x)$ . This is perhaps the simplest example of what is called ‘ $\omega$ -incompleteness’.

*Peano Arithmetic*, or PA, is the theory containing (1)–(6) and (8) from  $Q$  and all instances of the induction scheme:

$$A(0) \wedge \forall x (A(x) \rightarrow A(S(x))) \rightarrow \forall x A(x)$$

where  $A(x)$  is a formula of the language of arithmetic. Note that the standard interpretation of the language of arithmetic is a model of PA: All of the infinitely many axioms of PA are true in it.

**Proposition 3.3.** *Axiom (7) of  $Q$  is provable in PA.*

*Proof.* Exercise. □

*Arithmetic* is the theory containing all truths of arithmetic—that is, all formulae in the language of arithmetic that are true in the standard interpretation. Arithmetic is a complete, closed theory. It is complete, because every formula is either true or false in the standard interpretation; so for any formula  $A$ , either  $A$  is in arithmetic or its negation is. It is closed because, if  $A$  is derivable from arithmetic, it is true in every model of arithmetic; but since the standard interpretation *is* a model of arithmetic,  $A$  must be true in the standard interpretation, and so it is in arithmetic.

#### 4. REPRESENTABILITY

PA is sufficient to prove many facts about numbers. Consider, for example, the claim that  $x \neq x^2 + 1$ , for any  $x$ . Can that be proven in PA? One might worry that the symbol for *squared* is not part of the language of arithmetic, and one would be right to raise this issue. But it is no real obstacle, for there is an obvious way to define ‘ $x^2$ ’—as ‘ $x \times x$ ’—whence we can raise the question whether PA proves ‘ $x \neq (x \times x) + 1$ ’. By the obvious induction, it does. Similarly, PA can prove that  $x < x^2 + 1$ .

Consider now the question whether PA can prove that  $x < 2^x$ . The natural idea would again be to define exponentiation in the language of arithmetic, that is, to find some term  $\varphi(x)$  with just ‘ $x$ ’ free that defines  $2^x$ —defines it, in the sense that the value of  $\varphi(x)$ , when ‘ $x$ ’ is assigned the number  $n$ , is  $2^n$ . That, unfortunately, turns out to be impossible, though, as the following two results show.

**Proposition 4.1.** *Let  $\varphi(x)$  be a term with at most ‘ $x$ ’ free. Then  $\varphi(x)$  is equivalent to a polynomial in ‘ $x$ ’, that is, to something of the form:  $a_0x^n + a_1x^{n-1} + \dots + a_n$ .*

*Proof.* Induction on the complexity of expressions. Obviously, the basic terms ‘0’ and ‘ $x$ ’ are equivalent to polynomials. And if  $t$  and  $u$  are, then clearly ‘ $St$ ’, ‘ $t + u$ ’, and ‘ $t \times u$ ’ are as well. □

**Proposition 4.2.** *Let  $\varphi(x)$  be a polynomial. Then, for some  $y$ ,  $2^y > \varphi(y)$ .*

*Proof.* Exercise. □

It then follows, obviously, that if  $\varphi(x)$  is a term in the language of arithmetic containing just ‘ $x$ ’ free, then, for some  $n$ , the value of ‘ $\varphi(x)$ ’ when  $x$  is assigned  $n$  is less than, and so not equal to,  $2^n$ . So no term of the language of arithmetic defines exponentiation.

There is, however, a way around this problem, for there is a *formula*  $\text{EXP}(x,y)$  that defines exponentiation in a closely related sense:  $\text{EXP}(x,y)$  holds in the standard interpretation when ‘ $x$ ’ is assigned  $n$  and ‘ $y$ ’ is assigned  $m$  if, and only if,  $m = 2^n$ . So  $\text{EXP}(x,y)$  defines not the exponentiation *function* but the *relation*  $y = 2^x$ , and that surely is almost as good. Moreover,  $\text{EXP}(x,y)$  is, as Frege liked to put it, a ‘many-one’ relation: It can be proven in PA to be ‘function-like’, in the sense that  $\forall x \exists y (\text{EXP}(x,y) \wedge \forall z (\text{EXP}(x,z) \rightarrow y = z))$  can be proven in PA.<sup>10</sup> So we can think of PA as proving that  $x < 2^x$  if it proves

$$\forall x \exists y (\text{EXP}(x,y) \wedge x < y)$$

for what that says, in effect, is that, for each  $x$ , there is a  $y$ , such that  $y = 2^x$  and  $x < y$ . (And given an appropriate definition of  $\text{EXP}(x,y)$ , PA does prove that.)

These reflections motivate the following definitions.<sup>11</sup>

**Definition.** A formula  $\Phi(x_1, \dots, x_n, y)$  *defines* the  $n$ -place function  $\varphi(x_1, \dots, x_n)$  in the language of arithmetic if, and only if,

- (i)  $\forall x \exists y [\Phi(x_1, \dots, x_n, y) \wedge \forall z (\Phi(x_1, \dots, x_n, z) \rightarrow y = z)]$  is true (in the standard interpretation);
- (ii)  $\Phi(x_1, \dots, x_n, y)$  is true (in the standard interpretation), when  $x_i$  is assigned  $k_i$  and  $y$  is assigned  $m$  iff  $\varphi(k_1, \dots, k_n) = m$ .

We say that a function is *definable* in the language of arithmetic if there is a formula that defines it.

What we need here is a closely related notion that stands to definability much as the notion of derivation stands to that of implication.

**Definition.** A formula  $\Phi(x_1, \dots, x_n, y)$  *represents* the  $n$ -place function  $\varphi(x_1, \dots, x_n)$  in the theory  $\Sigma$  if, and only if, whenever  $\varphi(k_1, \dots, k_n) = m$ :

- (i)  $\Sigma \vdash \Phi(\mathbf{k}_1, \dots, \mathbf{k}_n, \mathbf{m})$
- (ii)  $\Sigma \vdash \forall y (\Phi(\mathbf{k}_1, \dots, \mathbf{k}_n, y) \rightarrow y = \mathbf{m})$

We say that a function is *representable* in  $\Sigma$  if there is a formula that represents it in  $\Sigma$ .

Note that the definition of representation requires only that  $\Sigma$  prove that  $\varphi$  has the value it does, *in each specific case*, and also that it prove that  $\varphi$  has only one value, *in each specific case*. It is *not* required that  $\Sigma$  prove the analogue of (i) in the definition of ‘defines’:

$$\forall x_1 \dots \forall x_n [\exists y (\Phi(x_1, \dots, x_n, y) \wedge \forall z (\Phi(x_1, \dots, x_n, z) \rightarrow y = z))].$$

It is, in fact, important that we omit this stronger condition. As we shall see below, it is true that  $Q$  represents exponentiation, that is, that there is a formula  $\text{EXP}(x,y)$  such that, whenever  $n = 2^m$ ,  $Q$  proves:

<sup>10</sup>Strictly speaking, of course, PA does not prove the mentioned formula, since ‘EXP’ is not in the language of arithmetic. What PA proves, strictly speaking, is  $\ulcorner \forall x \exists y (\text{EXP}(x,y) \wedge \forall z (\text{EXP}(x,z) \rightarrow y = z)) \urcorner$ , the result of substituting for the various occurrences of ‘EXP’ in the mentioned formula of the language of arithmetic that it abbreviates.

<sup>11</sup>The contrast between representability and definability here is *not* the same as the contrast between these notions in Boolos and Jeffrey (1989). There, definability is just the analogue of representability for sets and relations. Here, on the other hand, definability is a semantic notion; representability, a syntactic one. (My terminology is thus closer to Tarski’s.) The reason for this change will become clear below.

- (i)  $\text{EXP}(\mathbf{m}, \mathbf{n})$
- (ii)  $\forall y(\text{EXP}(\mathbf{m}, y) \rightarrow y = \mathbf{n})$

But it is an important fact about  $\mathcal{Q}$ —one that is, again, a reflection of its weakness—that it does *not* prove that exponentiation always has a unique value. It does not, that is, prove:

$$\forall x[\exists y(\text{EXP}(x, y) \wedge \forall z(\text{EXP}(x, z) \rightarrow y = z))]$$

and it does not prove that exponentiation is total:<sup>12</sup>

$$\forall x \exists y(\text{EXP}(x, y))$$

PA, however, proves both of these facts.

**Proposition 4.3.** *Suppose  $\Phi(x_1, \dots, x_n, y)$  represents a total function  $\varphi(x_1, \dots, x_n)$  in  $\Sigma$  and, whenever  $l \neq m$ ,  $\Sigma$  proves  $\ulcorner \mathbf{l} \neq \mathbf{m} \urcorner$ . Then whenever  $\Phi(x_1, \dots, x_n, y) \neq m$ ,  $\Sigma \vdash \neg\Phi(\mathbf{k}_1, \dots, \mathbf{k}_n, \mathbf{m})$ .*

*Proof.* Fix the  $k_i$  and suppose that  $\varphi(k_1, \dots, k_n) \neq m$ . For some  $l \neq m$ ,  $\varphi(k_1, \dots, k_n) = l$  and so, since  $\Phi(x_1, \dots, x_n, y)$  represents  $\varphi(x_1, \dots, x_n)$  in  $\Sigma$ ,  $\Sigma$  proves  $\forall y(\Phi(\mathbf{k}_1, \dots, \mathbf{k}_n, y) \rightarrow y = \mathbf{l})$ . So, by logic,  $\Sigma$  proves  $\Phi(\mathbf{k}_1, \dots, \mathbf{k}_n, \mathbf{m}) \rightarrow \mathbf{m} = \mathbf{l}$ . But now, by hypothesis,  $\Sigma$  proves  $\ulcorner \mathbf{l} \neq \mathbf{m} \urcorner$ , so it also proves  $\neg\Phi(\mathbf{k}_1, \dots, \mathbf{k}_n, \mathbf{m})$ .  $\square$

**Corollary 4.4.** *Suppose  $\Phi(x_1, \dots, x_n, y)$  represents a total function  $\varphi(x_1, \dots, x_n)$  in  $\mathcal{Q}$ . Then, if  $\varphi(k_1, \dots, k_n) \neq m$ ,  $\mathcal{Q} \vdash \neg\Phi(\mathbf{k}_1, \dots, \mathbf{k}_n, \mathbf{m})$ .*

*Proof.* Immediate from 3.2 and the preceding proposition.  $\square$

Similar notions can be defined for sets—or, more generally, relations—rather than functions.

**Definition.** A formula  $\Phi(x_1, \dots, x_n)$  *defines* the  $n$ -place relation  $F(x_1, \dots, x_n)$  if  $\Phi(x_1, \dots, x_n)$  is true (in the standard interpretation), when ‘ $x_i$ ’ is assigned  $k_i$ , iff  $F(k_1, \dots, k_n)$ . We say that a relation is *definable* in the language of arithmetic if there is a formula that defines it.

Slightly less formally:  $\Phi(x_1, \dots, x_n)$  *defines*  $F(x_1, \dots, x_n)$  if the extension of  $\Phi(x_1, \dots, x_n)$ , in the standard interpretation, is the set  $\{ \langle k_1, \dots, k_n \rangle : F(k_1, \dots, k_n) \}$ .

**Definition.** A formula  $\Phi(x_1, \dots, x_n)$  *represents* the  $n$ -place relation  $F(x_1, \dots, x_n)$  in the theory  $\Sigma$  iff:

- (i) whenever  $F(k_1, \dots, k_n)$ ,  $\Sigma \vdash \Phi(k_1, \dots, k_n)$
- (ii) whenever  $\neg F(k_1, \dots, k_n)$ ,  $\Sigma \vdash \neg\Phi(k_1, \dots, k_n)$

We say that a relation is *representable* in  $\Sigma$  if there is a formula that represents it in  $\Sigma$ .

These definitions are reconciled with those for functions through the notion of a *characteristic function*.

**Definition.** Let  $s$  be a set. Its *characteristic function* is the function  $\varphi_s(x)$  whose value is 1 if  $x \in s$  and 0 if  $x \notin s$ . Similarly, if  $r$  is an  $n$ -place relation, its characteristic function is the  $n$ -place function  $\varphi_r$  whose value, for arguments  $k_1, \dots, k_n$  is 1 if  $r(k_1, \dots, k_n)$  and 0 otherwise.

<sup>12</sup>Even  $IA_0$ , which adds induction for bounded formulae to  $\mathcal{Q}$ , does not prove that exponentiation is total.

**Exercise 4.5.** Show that a set or relation is definable (or representable in  $\Sigma$ ) iff its characteristic function is definable (or representable in  $\Sigma$ ).

As noted above, representability is the way we overcome the fact that the language of arithmetic is term poor. As also noted, however, a theory can represent a function without proving that the function so represented really is a function. For example,  $Q$  does not prove that exponentiation is either many-one or total. If a theory  $\Sigma$  *does* prove that exponentiation is both many-one and total, however—that is, if  $\Sigma$  proves both  $\forall x[\exists y(\text{EXP}(x, y) \wedge \forall z(\text{EXP}(x, z) \rightarrow y = z))]$  and  $\forall x\exists y(\text{EXP}(x, y))$ —then it is very much as if a function-symbol for exponentiation present in the language. The following result helps make it clear in what sense that is so.

**Theorem 4.6.** Suppose  $\Sigma$  represents a function  $\phi(x)$  by means of a formula  $F(x, y)$ .<sup>13</sup> Suppose further that  $\Sigma$  proves:

$$\begin{aligned} \forall x\forall y\forall z(F(x, y) \wedge F(x, z) \rightarrow y = z) \\ \forall x\exists y(F(x, y)) \end{aligned}$$

Expand the language of  $\Sigma$  by adding a new function symbol  $f(x)$  and let  $\Sigma_f$  be  $\Sigma$  plus the new axiom:

$$f(x) = y \equiv F(x, y)$$

Then  $\Sigma_f$  is a conservative extension of  $\Sigma$ .

*Proof.* Exercise, using 2.4. □

## 5. RECURSIVENESS AND REPRESENTABILITY IN $Q$

We are going to need to know, below, that certain functions and relations are indeed representable in  $Q$ . One way to prove that they are is, obviously, to exhibit formulae that represent them: That, in fact, is how Gödel proceeds in his proof of the incompleteness theorem. We, however, shall take a shortcut.

Certain functions are, in an intuitive sense, *computable* or *calculable*, in the sense that there is a method, algorithm, or procedure by means of which one can in principle determine, given some arguments for a function, what its value is for those arguments. For example, addition and multiplication are computable, in this sense: We all learned the relevant algorithms in elementary school. Properties of natural numbers, and relations among them, may, in the same sense, be computable. The relation:  $x$  divides  $y$  (without remainder), is intuitively computable: Long division is an algorithm by means of which one can decide whether one number divides another; it is, by the way, also an algorithm by means of which one can calculate the remainder upon dividing  $y$  by  $x$ , and so remainder is also computable.

There is a developed mathematical theory of computability, *recursion theory*, which was founded by Alonzo Church, Alan Turing, Kurt Gödel, and others.<sup>14</sup> This theory provides a definition of the notion of a ‘recursive function’ that makes rigorous the intuitive notion of a computable function. Indeed, there are *many* such definitions of the set of recursive functions, all of which are provably equivalent. That gives us reason to suppose that recursive functions form a natural kind.

<sup>13</sup>The extension of many-place functions is trivial.

<sup>14</sup>See Martin Davis, *The Undecidable*, for some of the early papers in the subject.

Church famously conjectured that the recursive functions are exactly the computable ones—a claim known as *Church's Thesis*. It is widely, but not universally, held that one cannot prove Church's Thesis rigorously: To do so, one would need a mathematical definition of the set of computable functions, and that is precisely what the notion of recursive function is supposed to be. (This is a form of the paradox of analysis.) No one, however, has ever exhibited a function that is computable in the intuitive sense but that is not recursive. Turing, for his part, offered a philosophical argument for Church's Thesis, namely, that his definition of the class of recursive functions constitutes a *philosophical analysis* of the intuitive notion of a computable function.

Church's Thesis is now very widely accepted. We shall assume it here. For our purposes, then, one can show a function to be recursive by exhibiting (or, more often, alluding to) an algorithm that computes it. So  $x$  divides  $y$ , for example, is recursive, since long division computes it. And so forth. Functions whose arguments and values are not natural numbers can be computable, or recursive, in much the same sense. We shall accept Church's Thesis for them, too. So, for example, the syntactic function: the negation of the expression  $x$ , is recursive: To compute it, one just writes down '¬' followed by a left parenthesis, the expression  $x$ , and a right parenthesis—and then one stares at the result.

As we saw above,  $Q$  is, in many respects, a very weak theory. We are now ready to see that, in another respect,  $Q$  is quite a powerful theory.

**Theorem 5.1.** Every recursive function is representable in  $Q$ .

*Proof.* See Boolos et al. (2002). □

This is the key result that makes Tarski's Theorem go. Its proof, however, is long and complex, and, so far as I can see, the details—though extremely interesting in themselves—throw little light upon Tarski's Theorem. So we shall just assume 5.1 here.

**Corollary 5.2.** Every computable function is representable in  $Q$ .

*Proof.* By Church's thesis. □

**Corollary 5.3.** Every computable set and relation is representable in  $Q$ .

*Proof.* By 4.5, a set  $s$  is representable in  $\Sigma$  iff its characteristic function  $\varphi_s$  is representable in  $\Sigma$ . But, if  $s$  is computable, then so is  $\varphi_s$ , for one can compute  $\varphi_s(n)$  simply by computing whether  $n \in S$  and then returning the answer: 1, if it is, and 0, if it is not. □

So it suffices to show that a function or set or relation is representable in  $Q$  to show that it is computable, that is, to exhibit (or allude to) an algorithm that computes it.

**Definition.** A theory  $\Theta$  extends a theory  $\Sigma$  if  $Cl(\Theta) \supseteq Cl(\Sigma)$ . We also say that  $\Theta$  contains  $\Sigma$ .

**Corollary 5.4.** Every computable recursive function is representable in every theory  $\Sigma$  that extends  $Q$ . Hence, so is every computable relation.

*Proof.* If a function is representable in  $Q$ , then it is representable in any theory that extends  $Q$ , since representability only requires that certain things should be theorems of the theory in question. □

## 6. GÖDEL NUMBERING

Our goal, you may recall, is to prove results about truth. One might wonder, therefore, why we have been talking about  $Q$  and other arithmetical theories. Isn't it just obvious that one can not produce a materially adequate theory of truth for the language of arithmetic in  $Q$ , or even in PA? Such a theory would have to have theorems like:

$$\text{TRUE}('SS0 = S0') \equiv (SS0 = S0)$$

But while it's clear enough what sort of role TRUE is going to play here—the same sort of role 'EXP' played above—it's far from clear what to do about the expression “ ‘SS0 = S0’ ” that occurs as its argument. Theories of truth are, obviously, theories about sentences and other linguistic expressions, and the language of arithmetic provides us with no resources to talk about sentences. Just as we found a way around the problem posed by arithmetic's lack of terms, though, there is a way around this problem, too: It is called *Gödel numbering*.

Consider the following correspondence between the primitive symbols of the language of arithmetic and the hexadecimal (base-16) digits:

(	1
)	2
∃	3
∀	4
∨	5
∧	6
→	7
¬	8
$x$	9
'	a
0	b
$S$	c
+	d
×	e
=	f

We extend the correspondence to one between strings of symbols of the language of arithmetic and hexadecimal numerals in the obvious way, basically, by treating these symbols as if they just *were* the corresponding hexadecimal digits. Thus, for example, we read the string ‘)+0∃∨’ as if it were the hexadecimal numeral ‘2db35’. This induces a correlation between strings of symbols and natural numbers. The one-element string ‘0’ is thus correlated with the number  $b_{16}$ , or 11; the string ‘)+0∃∨’ is correlated with the number  $2db35_{16}$ , or  $2 \times 16^4 + 14 \times 16^3 + 11 \times 16^2 + 3 \times 16^1 + 5 \times 16^0$ , or 191,285. Every string of symbols is thus correlated with its own unique natural number, called its *Gödel number*. Note that a natural number is the Gödel number of a string if and only if its hexadecimal representation contains no (non-leading) zeros. It is thus intuitively computable whether a natural number is the Gödel number of a string.

There are many ways to think of this correspondence between strings and numbers. For our purposes here, however, we shall adopt the one that makes the proofs that follow easiest to understand: We shall *identify* strings of symbols with the corresponding numbers. For our purposes here, the

string ‘ $\text{)}+0\exists\forall$ ’ *just is* the number 191,285. We shall also identify a one-element string with the single symbol of which it is composed. So ‘0’ *just is* the natural number 11.

For most purposes, the specific Gödel numbering we employ does not matter very much. What certainly does matter is that it should be computable: It should be computable what the Gödel number of a given string is; conversely, it needs to be computable whether a given number is the Gödel number of a string and, if so, of which one. Note that our Gödel numbering is computable, in this sense.

Gödel numbering allows us now to talk about syntax—about properties of strings, or sequences thereof, and operations on them—within the language of arithmetic. Consider, for example, the function *paren*, whose value, for a given string, is the result of enclosing that string in parentheses. Thus, *paren*[‘ $\text{)}+0\exists\forall$ ’] is just ‘ $\text{()}+0\exists\forall$ ’). Since, however, strings just are natural numbers in disguise, *paren* is actually a number theoretic function: Its value for ‘ $\text{)}+0\exists\forall$ ’—that is,  $2\text{db}35_{16}$ —is ‘ $\text{()}+0\exists\forall$ ’—that is,  $12\text{db}352_{16}$ . Indeed, one can describe this function directly:

$$\begin{aligned} \text{paren}(x) &= 0, \text{ if } x \text{ has (non-leading) zeros in its hexadecimal representation;} \\ &= 1 \times 16^{k+1} + 16 \times x + 2, \text{ otherwise, where } k \text{ is the least } n \text{ such that } 16^n > x \end{aligned}$$

Here 0 is a ‘throwaway’ value, provided so that *paren* is a total function. Since *paren*( $x$ ) is clearly computable, it is representable in  $Q$ . So there is a formula  $\text{PAREN}(x,y)$  such that, whenever  $n = \text{paren}(m)$ ,  $Q$  proves:  $\text{PAREN}(\mathbf{m}, \mathbf{n})$ . In particular, since  $\text{paren}(2\text{db}35_{16}) = 12\text{db}352_{16}$ ,  $Q$  proves  $\text{PAREN}(2\text{db}35_{16}, 12\text{db}352_{16})$ . Informally:  $Q$  proves that the result of enclosing ‘ $\text{)}+0\exists\forall$ ’ in parentheses is ‘ $\text{()}+0\exists\forall$ ’).

In GödelLand, they write their hexadecimal numerals with symbols different from those that we use. So, instead of: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d, e, and f, they use:  $\text{/}$ ,  $\text{(}$ ,  $\text{)}$ ,  $\exists$ ,  $\forall$ ,  $\vee$ ,  $\wedge$ ,  $\rightarrow$ ,  $\neg$ ,  $x$ ,  $\prime$ , 0,  $S$ ,  $+$ ,  $\times$ , and  $=$ . Where we would write:  $2\text{db}35_{16}$ , they write:  $\text{)}+0\exists\forall$ . In GödelLand, then, they would write the last two sentences of the foregoing paragraph as:

In particular, since  $\text{paren}[\text{)}+0\exists\forall] = \text{()}+0\exists\forall$ ,  $Q$  proves:  $\text{PAREN}[\text{)}+0\exists\forall, \text{()}+0\exists\forall]$ .  
That is:  $Q$  proves that the result of enclosing ‘ $\text{)}+0\exists\forall$ ’ in parentheses is ‘ $\text{()}+0\exists\forall$ ’).

It may well prove convenient to speak like the GödelLanders, from time to time.

Generalizing the above, we have the following.

**Lemma 6.1.** *Let  $\varphi(s_1, \dots, s_n)$  be a computable  $n$ -place function on strings. Then there is a formula  $\Phi(x_1, \dots, x_n, y)$  of the language of arithmetic that represents it in  $Q$ , in the sense that, whenever  $\varphi(s_1, \dots, s_n) = t$  and  $\sigma_1, \dots, \sigma_n$  and  $\tau$  are the Gödel numbers of  $s_1, \dots, s_n$  and  $t$ , respectively:*

- (i)  $Q \vdash \Phi(\sigma_1, \dots, \sigma_n, \tau)$
- (ii)  $Q \vdash \forall x (\Phi(\sigma_1, \dots, \sigma_n, x) \rightarrow x = \tau)$

Note that the Lemma has been stated, and will be proved, in such a way that any it holds for all computable Gödel numberings.

*Proof.* Let  $\varphi$  be as in the statement of the Lemma. We define a number-theoretic function  $\varphi^*$  as follows.

$$\begin{aligned}\varphi^*(x_1, \dots, x_n) &= \text{the Gödel number of } \varphi(s_1, \dots, s_n), \\ &\quad \text{if each } x_i \text{ is the Gödel number of a string } s_i \\ &= 0, \text{ otherwise}\end{aligned}$$

The definition of  $\varphi^*$  clearly makes it computable: To calculate  $\varphi^*(x_1, \dots, x_n)$ , one need only follow these steps: First, determine of which string  $s_i$  each of the  $x_i$  is the Gödel number, if any (there is a method for doing that, since the Gödel numbering is computable); return 0 if one of the  $x_i$  isn't the Gödel number of a string; otherwise, calculate  $\varphi(s_1, \dots, s_n)$  and then determine its Gödel number (there are methods for doing each of those things, since  $\varphi$  is computable and so is the Gödel numbering); hence,  $\varphi^*$  is computable and so is representable in  $Q$ . So there is a formula  $\Phi(x_1, \dots, x_n, y)$  such that, whenever  $\varphi^*(\sigma_1, \dots, \sigma_n) = \tau$ :

- (i)  $Q \vdash \Phi(\sigma_1, \dots, \sigma_n, \tau)$
- (ii)  $Q \vdash \forall x (\Phi(\sigma_1, \dots, \sigma_n, x) \rightarrow x = \tau)$

But if, as in the statement of the lemma,  $\varphi(s_1, \dots, s_n) = t$  and  $\sigma_1, \dots, \sigma_n$  and  $\tau$  are the Gödel numbers of  $s_1, \dots, s_n$  and  $t$ , respectively, then we do indeed have that  $\varphi^*(\sigma_1, \dots, \sigma_n) = \tau$ , by the definition of  $\varphi^*$ .  $\square$

**Corollary 6.2.** *Let  $\varphi(s_1, \dots, s_n)$  be a computable  $n$ -place relation on strings. Then there is a formula  $\Phi(x_1, \dots, x_n)$  of the language of arithmetic that represents it in  $Q$ , in the sense that, where  $\sigma_1, \dots, \sigma_n$  are the Gödel numbers of  $s_1, \dots, s_n$ :*

- (i) *whenever  $\varphi(s_1, \dots, s_n)$ ,  $Q \vdash \Phi(\sigma_1, \dots, \sigma_n)$*
- (ii) *whenever  $\neg\varphi(s_1, \dots, s_n)$ ,  $Q \vdash \neg\Phi(\sigma_1, \dots, \sigma_n)$*

*Proof.* Exercise.  $\square$

Many syntactic notions are now easily shown to be representable in  $Q$  by appeal to the preceding results. Thus, it is obviously computable whether a string of symbols of the language of arithmetic is a well-formed formula. Hence, there is a formula  $WFF(x)$  of the language of arithmetic that represents 'x is a well-formed formula'. Similarly, the relation 'z is a conditional whose antecedent is x and consequent is y' is obviously computable and hence representable in  $Q$ . And so on and so forth.

## 7. FORMAL THEORIES

**Definition.** A *formal theory* is a theory—that is, a set of formulae—that is recursive. The formulae in a formal theory are called its *axioms*.

Every finite theory is obviously a formal theory. PA is, too: There is an obvious algorithm by means of which one can decide, or compute, whether an arbitrary formula is an axiom of PA. It follows from Gödel's First Incompleteness Theorem, however, that Arithmetic is *not* a formal theory.

**Definition.** A theory  $\Sigma$  is said to be *axiomatizable* if there is a formal theory  $\Theta$  in the same language such that, for every formula  $A$  in their common language:  $\Sigma \vdash A$  iff  $\Theta \vdash A$ .

There is an equivalent statement of this definition. Given a theory  $\Sigma$ , define its *closure* as follows:

**Definition.** The closure of a theory  $\Sigma$ — $Cl(\Sigma)$ —is the set of its theorems, that is, of the formulae derivable from it.

We can easily prove that  $Cl(\Sigma)$  is always closed, justifying the terminology, and, moreover, that it is the smallest closed theory containing  $\Sigma$  itself. The equivalent definition of axiomatizability is then:

*Remark 7.1.*  $\Sigma$  is axiomatizable if there is a formal theory  $\Theta$  such that  $Cl(\Sigma) = Cl(\Theta)$ .

## 8. THE DIAGONAL LEMMA

We now prove the diagonal lemma, which is the last result we need to prove Tarski's Theorem.

First, a piece of notation. We shall use:  $\ulcorner G \urcorner$ , to mean: the numeral for the Gödel number of the expression  $G$ . We shall also use it to mean: the Gödel number of  $G$ , leaving it to context to disambiguate.

**Lemma 8.1.** (*The Diagonal Lemma*) *Let  $\Sigma$  be a theory that represents every recursive function; let  $A(x)$  be a formula in the language of  $\Sigma$  with just ' $x$ ' free. Then there is a sentence  $G$  such that  $\Sigma \vdash G \equiv A(\ulcorner G \urcorner)$ .*

Thus,  $G$  (for Gödel) is a sentence that 'says of itself' that it has the property that  $A(x)$  expresses. In this sense, the existence of self-referential sentences is mathematically provable: It is, more precisely, guaranteed by the axioms of  $\mathcal{Q}$ , since these guarantee that  $\mathcal{Q}$  satisfies the hypothesis of the diagonal lemma.

We shall prove the lemma two ways. We begin, however, with some motivational remarks.

Consider the following open term (i.e., functional expression):

the result of substituting the quotation name of  $x$  for the sole free variable contained in it.

If we apply this function to 'frogs eat  $x$ ', then we get:

frogs eat 'frogs eat  $x$ ',

which is presumably false. If we apply it to ' $x$  contains 16 symbols, including blanks', then the result is:

'x contains 16 symbols, including blanks' contains 16 symbols, including blanks,

which is also false. And if we apply it to ' $x$  contains the word "the"', we get:

'x contains the word "the"' contains the word "the",

which is obviously true.

Consider now what happens if we apply the function to 'the result of substituting the quotation name of  $x$  for the sole free variable contained in it is very strange'. The result is:

the result of substituting the quotation name of ‘the result of substituting the quotation name of  $x$  for the sole free variable contained in it is very strange’ for the sole free variable contained in it is very strange

Call this sentence Buster. Then what we have just seen—proven, really, by calculation—is that Buster is the result of substituting the quotation name of ‘the result of substituting the quotation name of  $x$  for the free variable contained in it is very strange’ for the sole free variable contained in it. So, by the laws of identity, Buster is equivalent to the sentence: Buster is very strange. In that sense, then, Buster ‘says of itself’ that it is very strange.

The formal proof we shall give in a bit is just a formalization of this informal argument. It is, however, in some ways more complicated than it really ought to be. The complications are the result of the fact that, as we saw above, the language of arithmetic is rather lacking in terms. If we ignore that and pretend there are a lot of terms, we can give a much simpler proof that makes the *point* of the argument a lot clearer.

*Proof. (Somewhat dishonest variety)*<sup>15</sup> Consider the following syntactic operation: Given a formula  $A(x)$ , with just ‘ $x$ ’ free, the value of the function—which we shall call the pseudo-diagonalization of  $A(x)$ —is the result of substituting the numeral for the Gödel number of  $A(x)$  for ‘ $x$ ’ in  $A(x)$ . That is, the pseudo-diagonalization of  $A(x)$  is:  $A(\ulcorner A(x) \urcorner)$ . This operation is obviously recursive, so (we will pretend) there is an open term,  $\text{DIAG}(x)$ , that ‘represents’ pseudo-diagonalization in  $\Sigma$ , in the sense that, whenever  $n$  is the Gödel number of the pseudo-diagonalization of the expression whose Gödel number is  $m$ ,  $\Sigma$  proves  $\text{DIAG}(\mathbf{m}) = \mathbf{n}$ . Now consider the expression:<sup>16</sup>

$$(P) \quad A(\text{DIAG}(x)),$$

which has only the variable ‘ $x$ ’ free. Its pseudo-diagonalization is then:

$$(G) \quad A(\text{DIAG}(\ulcorner A(\text{DIAG}(x) \urcorner) \urcorner)),$$

which is a sentence.<sup>17</sup> So  $\ulcorner G \urcorner$  is the Gödel number of a formula—namely,  $G$ —which is the pseudo-diagonalization of a formula—namely,  $P$ —whose Gödel number is  $\ulcorner A(\text{DIAG}(x) \urcorner)$ . Hence:

$$\Sigma \vdash \text{DIAG}(\ulcorner A(\text{DIAG}(x) \urcorner) \urcorner) = \ulcorner G \urcorner$$

and so, by the laws of identity:

$$\Sigma \vdash A(\text{DIAG}(\ulcorner A(\text{DIAG}(x) \urcorner) \urcorner)) \equiv A(\ulcorner G \urcorner)$$

But the sentence on the left-hand side here *just is*  $G$ , so what we have just seen is:

$$\Sigma \vdash G \equiv A(\ulcorner G \urcorner)$$

So we are done. □

<sup>15</sup>There are ways one can make this argument legitimate, if  $\Sigma$  is strong enough. PA is ‘strong enough’. So is  $I\Sigma_1$ , which is  $Q$  with induction for  $\Sigma_1$  sentences. See *The Logic of Provability* (Boolos, 1993), from which this proof is adapted, for the details. Moreover, there are systems, such as primitive recursive arithmetic, in which we really do have a function symbol such as  $\text{DIAG}(x)$ , and in such systems the proof can stand as is.

<sup>16</sup>Note that  $\ulcorner A(\text{DIAG}(x) \urcorner)$  is the result of substituting  $\text{DIAG}(x)$  for all free occurrences of ‘ $x$ ’ in  $A(x)$ .

<sup>17</sup>Note carefully that ‘ $x$ ’ does not occur in (G). It occurs only ‘quoted’.

The ‘real’ proof of the diagonal lemma we shall now give just recapitulates this argument, but it makes use of a *formula* ‘DIAG( $x, y$ )’ instead of the pretend term ‘DIAG( $x$ )’: As said earlier, this sort of complication is the price we pay for the language of arithmetic’s poverty of terms. We first need a definition.

**Definition.** Let  $A$  be an expression. Its *diagonalization* is the expression:  $\exists x(x = \ulcorner A \urcorner \wedge A)$ , where  $\ulcorner A \urcorner$  is, as before, the numeral for the Gödel number of  $A$ .

The expression  $A$  need not be a (well-formed) formula. If it isn’t, then its diagonalization will not be a formula either. But if  $A$  is a formula, then its diagonalization will also be a formula, and, if  $A$  has only the variable ‘ $x$ ’ free, then its diagonalization will be a *closed* formula, that is, a sentence.

You will note that all we *really* need to know for the proof of the diagonal lemma is that diagonalization is representable in  $\Sigma$ . So what we will actually prove is

**Lemma 8.2.** (*Diagonal Lemma*) *Let  $\Sigma$  be a theory that represents diagonalization. Then, for each formula  $A(x)$  with just ‘ $x$ ’ free, there is a formula  $G$  such that  $\Sigma \vdash G \equiv A(\ulcorner G \urcorner)$ .*

*Proof.* Since  $\Sigma$  represents diagonalization, there is a formula  $\text{DIAG}(x, y)$ <sup>18</sup> such that, whenever  $n$  is the Gödel number of the diagonalization of the expression whose Gödel number is  $m$ ,  $\Sigma \vdash \text{DIAG}(\mathbf{m}, \mathbf{n})$  and  $\Sigma \vdash \forall x(\text{DIAG}(\mathbf{m}, x) \rightarrow x = \mathbf{n})$ . Now, consider the expression:

$$(P) \quad \exists y(\text{DIAG}(x, y) \wedge A(y)),$$

which has only the variable ‘ $x$ ’ free. Let its Gödel number be  $p$ . Its diagonalization is then:

$$(G) \quad \exists x[x = \mathbf{p} \wedge \exists y(\text{DIAG}(x, y) \wedge A(y))].$$

Note that  $G$  is a sentence. Let its Gödel number be  $g$ .

Since  $G$  is logically equivalent to:

$$\exists y[\text{DIAG}(\mathbf{p}, y) \wedge A(y)],$$

$\Sigma$  certainly proves:

$$(1) \quad \exists x[x = \mathbf{p} \wedge \exists y(\text{DIAG}(x, y) \wedge A(y))] \equiv \exists y[\text{DIAG}(\mathbf{p}, y) \wedge A(y)].$$

Further,  $G$ , as we said, is the diagonalization of  $P$ . So, obviously,  $g$  is the Gödel number of the diagonalization of the formula (namely,  $G$ ) whose Gödel number is  $p$  (namely,  $P$ ). Hence  $\Sigma$  proves:

$$\begin{array}{l} \text{DIAG}(\mathbf{p}, \mathbf{g}) \\ \forall x[\text{DIAG}(\mathbf{p}, x) \rightarrow x = \mathbf{g}] \end{array}$$

or, more simply:

$$(2) \quad \forall x[\text{DIAG}(\mathbf{p}, x) \equiv x = \mathbf{g}]$$

From (2),  $\Sigma$  will prove:

$$\exists y[\text{DIAG}(\mathbf{p}, y) \wedge A(y)] \equiv \exists y[y = \mathbf{g} \wedge A(y)]$$

but the right-hand side is obviously equivalent to ‘ $A(\mathbf{g})$ ’, so  $\Sigma$  proves:

---

<sup>18</sup>It is worth emphasizing that  $\text{DIAG}$  is a formula that can actually be written down. George Boolos told me that he once had a graduate student write it out in primitive notation. It was about six pages long.

$$(3) \quad \exists y[\text{DIAG}(\mathbf{p}, y) \wedge A(y)] \equiv A(\mathbf{g})$$

And now, from (1) and (3),  $\Sigma$  proves:

$$(4) \quad \exists x[x = \mathbf{p} \wedge \exists y(\text{DIAG}(x, y) \wedge A(y))] \equiv A(\mathbf{g})$$

But the left-hand side here is just  $G$ , and  $g$  is the Gödel number of  $G$ . So (4) just is:

$$G \equiv A(\ulcorner G \urcorner),$$

as wanted. □

Of course, the diagonal lemma, as we originally stated it, is now easily provable. Since diagonalization is clearly computable, it is representable in any theory that represents all computable functions. Any such theory therefore satisfies the hypothesis of the diagonal lemma.

The diagonal lemma can be generalized in two ways: We can allow other free variables in  $A$ , which just get carried along; and we can apply the construction simultaneously to a number of formulae.

**Lemma 8.3.** *(The Generalized Diagonal Lemma) Let  $\Sigma$  be a theory that represents every recursive function and, for  $i=1, \dots, n$ , let  $A_i(x_1, \dots, x_n, y_1, \dots, y_m)$  be formulae in the language of  $\Sigma$  with at most the displayed variables free. Then there are formulae  $G_1, \dots, G_n$  such that, for each  $i$ ,*

$$\Sigma \vdash G_i(y_1, \dots, y_m) \equiv A_i(\ulcorner G_1 \urcorner, \dots, \ulcorner G_n \urcorner, y_1, \dots, y_m)$$

The proof is a generalization of that of the diagonal lemma and is left as an exercise (which does not mean it is easy).

## 9. TARSKI'S THEOREM

We now prove Tarski's Theorem.

**Lemma 9.1.** *(The Liar Paradox) Let  $\Sigma$  be a theory in which diagonalization is representable and so for which the diagonal lemma holds. Suppose that there is a formula  $T(x)$  of the language of  $\Sigma$  such that, for every sentence  $A$  of the language of  $\Sigma$ :*

$$(T) \quad \Sigma \vdash A \equiv T(\ulcorner A \urcorner)$$

*Then  $\Sigma$  is inconsistent.*

*Proof.* Let  $T(x)$  be as in the statement of the theorem. Consider:  $\neg T(x)$ . By the diagonal lemma, there is a sentence  $\lambda$  such that  $\Sigma$  proves:

$$(1) \quad \lambda \equiv \neg T(\ulcorner \lambda \urcorner).$$

But since  $\Sigma$  proves all instances of the T-schema, we have that  $\Sigma$  proves:

$$(2) \quad \lambda \equiv T(\lambda).$$

But if  $\Sigma$  proves both (1) and (2), it proves

$$\lambda \equiv \neg \lambda$$

and so is (classically) inconsistent. □

There is another way of stating this result. First, we need a definition.

**Definition.** A theory  $\Sigma$  contains its own truth-predicate if there is a formula  $T(x)$  of the language of  $\Sigma$  such that, for every sentence  $A$  of that language,  $\Sigma$  proves:  $A \equiv T(\ulcorner A \urcorner)$ .

The restatement is then as follows.

**Corollary 9.2.** *Let  $\Sigma$  be a consistent theory in which diagonalization is representable. Then  $\Sigma$  does not contain its own truth-predicate.*

*Proof.* Immediate from the Liar Paradox. □

Note that it is not assumed that  $\Sigma$  is a formal theory. So we get:

**Corollary 9.3.** *Arithmetic does not contain its own truth-predicate. That is, there is no formula  $T(x)$  of the language of arithmetic such that all instances of schema (T) are true (in the standard interpretation).*

*Proof.* Arithmetic is the theory whose members are all the sentences of the language of arithmetic that are true in the standard interpretation. As we saw earlier, arithmetic is closed: So its theorems are just its members. And so, for every sentence  $B$  of the language of arithmetic,  $B$  is true in the standard interpretation iff *Arithmetic*  $\vdash B$ .

Now, suppose there is a formula  $T(x)$  in the language of arithmetic such that, for every sentence  $A$  of that language, ' $A \equiv T(\ulcorner A \urcorner)$ ' is true in the standard interpretation. Then, for every sentence  $A$ , *Arithmetic*  $\vdash A \equiv T(\ulcorner A \urcorner)$ . But arithmetic extends  $Q$  and so represents all recursive functions and so represents diagonalization. So that contradicts corollary 9.2, since arithmetic is consistent. □

**Theorem 9.4.** (Tarski's Theorem, Special Case) *Arithmetical truth is not arithmetically definable. That is: There is no formula  $T(x)$  of the language of arithmetic such that  $T(\mathbf{n})$  is true in the standard interpretation when and only when  $\mathbf{n}$  is the Gödel number of a sentence of the language of arithmetic that is itself true in the standard interpretation.*

*Proof.* Suppose there were such a formula. Let  $A$  be arbitrary.  $A$  is either true or false in the standard interpretation. If it is true, then  $T(\ulcorner A \urcorner)$  must be true in the standard interpretation, whence ' $A \equiv T(\ulcorner A \urcorner)$ ' is true, as well. Similarly, if  $A$  is false, then  $T(\ulcorner A \urcorner)$  is false and so ' $A \equiv T(\ulcorner A \urcorner)$ ' is again true. So all instances of schema (T) are true, contradicting corollary 9.3. □

Here's an application of the generalized diagonal lemma. Suppose  $\Sigma$  represents diagonalization. Let  $A_1(x_1, x_2)$  be:  $\neg T(x_2)$ , and let  $A_2(x_1, x_2)$  be:  $T(x_1)$ . By the generalized diagonal lemma, there are sentences  $\lambda$  and  $\mu$  such that  $\Sigma$  proves:

- (a)  $\lambda \equiv \neg T(\ulcorner \mu \urcorner)$
- (b)  $\mu \equiv T(\ulcorner \lambda \urcorner)$

Suppose now that  $\Sigma$  proves:

- (c)  $\lambda \equiv T(\ulcorner \lambda \urcorner)$
- (d)  $\mu \equiv T(\ulcorner \mu \urcorner)$

Then  $\Sigma$  is inconsistent. For we have  $\lambda \equiv_{(c)} T(\ulcorner \lambda \urcorner) \equiv_{(b)} \mu \equiv_{(d)} T(\ulcorner \mu \urcorner) \equiv_{(a)} \neg\lambda$ , where the subscripts indicate to which of (a)–(d) we are appealing.

This is known as the Postcard Paradox: Imagine a postcard on one side of which is written “The sentence on the other side of this card is false”; on the other side is written “The sentence on the other side of this card is true”.

#### REFERENCES

- Boolos, G. (1993). *The Logic of Provability*. Cambridge University Press, New York.
- Boolos, G. S., Burgess, J. P., and Jeffrey, R. C. (2002). *Computability and Logic*, 4th edition. Cambridge University Press, Cambridge.
- Boolos, G. S. and Jeffrey, R. C. (1989). *Computability and Logic*, 3d edition. Cambridge University Press, New York.