# Class Notes for CSI 530 on the Incompleteness Theorem

*Dan E. Willard* [*]

March 2004 - Revised April 2010 with Löb Appendix Added

**Goal of these Notes:** These lecture notes are intended to summarize the intuition behind Gödel's First and Second Incompleteness Theorem. I believe that they should be easier to comprehend than Mendelson's treatment [3] of the subject. They assume the reader is familiar with Sections 1.1 through 1.4, Sections 2.1 through 2.6 and Section 3.1 of his textbook.

These notes are not intended as a replacement for the Sections 3.2 through 3.5 of Mendelson's textbook. Rather they are intended to prove a slightly weaker theorem that captures the main intuition behind those sections in a small amount of page space.

You are the **first class** to receive these lecture notes. Therefore, I ask you to please proof-read them, and let me know what typos you have found and which points confuse you? For example, if I did not carefully define some construct that you wish to understand, please let me know what it is ?

These lecture notes will be improved by your feedback.

[*]Address: Dep of CS, SUNYA, Albany, NY 12222 or dew@cs.albany.edu.

## 1 Introduction

Our language $L$ in these notes will include four symbols $C_0$ , $C_1$ , $C_2$ and $C_3$ , representing the integers of 0, 1, 2 and 3. The language $L$ will discuss the properties of non-negative integers. It will define all integers larger than 3 implicitly in terms of these base integers.

The predicate symbols used by our language $L$ will be the equality and less-than-or-equal predicates, denoted as " $=$ " and " $\leq$ " . Sometimes, we will informally also use the symbols $\geq$ , $<$ and $>$ .

Define $F(a_1, a_2, ...a_j)$ to be a **NON-GROWTH FUNCTION** iff for all values of $a_1, a_2, ...a_j$, the function $F$ satisfies $F(a_1, a_2, ...a_j) \leq Maximum(a_1, a_2, ...a_j)$ . Our axiom systems will employ a set of eight non-growth functions, called the **GROUNDING FUNCTIONS**. They will include:

1. Integer Subtraction where $x - y$ is defined to equal zero when $x < y$,
2. Integer Division where $\frac{x}{y} = x$ when $y = 0$ , and it otherwise equals $\lfloor \frac{x}{y} \rfloor$ .
3. $Predecessor(x) = \text{Max}(\, x - 1 \, , \, 0 \,)$,
4. $Maximum(x, y)$,
5. $Logarithm(x) = \lceil Log_2(x + 1) \rceil$,
6. $Root(x, y) = \lfloor x^{1/y} \rfloor$ when $y \geq 1$ , and $Root(x, 0) = x$.
7. $Count(x, j)$ designating the number of "1" bits among $x$'s rightmost $j$ bits.
8. $Bit(x, i)$ designating the value of the integer $x$'s $i-$th rightmost bit. (Note that $Bit(x, i) = Count(x, i) - Count(x, i - 1)$.)

In addition to the above non-growth functions, our language $L$ will employ two growth functions. They will correspond to Integer-Addition and Double$(x) = x + x$. We will use the term **U-Grounding Function** to refer to a function that is one of our eight Grounding Functions or the operations of Addition and Doubling.

**Comment 1.1.** We do not technically need both the operations of Addition and Doubling in our U-Grounding language $L$. However, it much is easier to encode large integers when we have access to both these function symbols. For example, any integer $N > 1$ can be encoded by a term of length $O(\text{ Log}(N) )$, using only the constant symbol for "1", when both the addition and doubling function symbols are present. For instance, below is our binary-like encoding for the number 11

$$1 \; + \; \text{Double}( \; 1 + \; \text{Double}( \; \text{Double}( \; 1))) \tag{1}$$

Henceforth, the symbol $\overline{N}$ will denote such a binary-like encoding for the integer $N$. (In the degenerate case where $N = 0$, $\overline{0}$ will simply be defined as being the constant symbol " $C_0$ ", that represents zero's value.)

**Definition 1.2.** We will follow mostly conventional logic notation when discussing U-Grounding functions. Thus, a *term* is defined to be a constant symbol, a variable symbol or a function symbol (followed by some input arguments, which are similarly defined terms). If $t$ is a term then the quantifiers in $\forall \; v \leq t \;\; \Psi(v)$ and $\exists \; v \leq t \;\; \Psi(v)$ will be called *bounded quantifiers.* These two wffs will be semantically equivalent to the formulae of $\forall v \;\; ( \; v \leq t \; \Rightarrow \; \Psi(v))$ and $\exists v \;\; ( \; v \leq t \wedge \Psi(v))$. A formula $\Phi$ will be called $\Delta_0^*$ iff all its quantifiers are bounded. Thus a $\Delta_0^*$ formula is defined to be a wff that is **any combination** of atomic formulae (using our ten U-Grounding functions and the equals and $\leq$ predicates) combined by bounded quantifiers and the boolean operations of AND,OR, NOT and IMPLIES in an arbitrary manner.

**Example 1.3** Below are four examples of $\Delta_0^*$ formulae.

$$\forall x \leq \; \overline{10} \quad \forall y \leq 20 \quad x + y = y + x \tag{2}$$

$$\forall x \leq \; \overline{100} \quad \exists y \leq \; \overline{60} \quad x \leq \text{Double}(y) \tag{3}$$

$$\forall x \leq \; \overline{10} \qquad \forall y \leq [\, \overline{50} \; + \text{Double}(x)] \qquad [ \quad x + y = 0 \quad \Rightarrow \quad x = 0 \; \wedge y = 0 \; ] \tag{4}$$

$$\forall x \leq \; \overline{10} \quad x + 1 = x \tag{5}$$

Note that Equation (5) is defined to be a $\Delta_0^*$ formulae (even though it is a false statement). The point is that $\Delta_0^*$ formulae can be either true or false. Their definition differs from Logic's conventional formulae only by requiring that all quantifiers be "bounded". Also a $\Delta_0^*$ formula does not need to contain any quantifiers (e.g. Equations (6) and (7) are also $\Delta_0^*$ formulae).

$$\overline{10} + \overline{25} \;\geq\; \overline{30} \tag{6}$$

$$\overline{10} + \overline{25} \;=\; \overline{30} \quad \vee \quad \overline{4} + \overline{7} \;=\; \overline{11} \tag{7}$$

Finally, let $F_1$ through $F_6$ denote the six $\Delta_0^*$ formulae above. Then since a $\Delta_0^*$ formula can be any Boolean combination of bounded quantifiers with the "boolean operations" of AND,OR, NOT and IMPLIES, some other examples of $\Delta_0^*$ formula are:

$$F_1 \;\wedge\; F_2 \;\wedge\; (\; F_6 \;\vee \neg\, F_5\;) \tag{8}$$

$$F_2 \;\wedge\; F_3 \Rightarrow F_4 \tag{9}$$

Once again, we emphasize that a $\Delta_0^*$ formula can be either true or false. For instance, (8) is true while (9) is false.

**Definition 1.4** For any integer $i \geq 0$ , this paragraph will define the notions of a $\Pi_i^*$ and $\Sigma_i^*$ formula. The definition has three parts and it is given below.

1. Every $\Delta_0^*$ formula is defined to be also both a $\Pi_0^*$ and $\Sigma_0^*$ formula.
2. A formula is called $\Pi_{i+1}^*$ iff for some $\Sigma_i^*$ formula $\Phi(v_1, v_2, ...v_n)$, it can be written in the form $\forall v_1\ \forall v_2\ ...\ \forall v_n\ \Phi(v_1, v_2, ...v_n)$ . (Since this rule also applies when the number of quantifiers $n$ equals zero, it follows that every $\Sigma_i^*$ formula is automatically by default also $\Pi_{i+1}^*$.)
3. Similarly, a formula is called $\Sigma_{i+1}^*$ iff for some $\Pi_i^*$ formula $\Phi(v_1, v_2, ...v_n)$, it can be written in the form $\exists v_1\ \exists v_2\ ...\ \exists v_n\ \Phi(v_1, v_2, ...v_n)$ . (Since this rule also applies when the number of quantifiers $n$ equals zero, it follows that every $\Pi_i^*$ formula is automatically by default also $\Sigma_{i+1}^*$.)

**Example 1.5.** Equations (10) are (11) are examples of a $\Pi_2^*$ sentences, and Equations (12) are (13) are examples of $\Pi_1^*$ sentences. Note that some $\Pi_2^*$ sentences can be proven to be logically equivalent to $\Pi_1^*$ sentences. Thus for example, the sentences in Equations (10) and (13) are logically equivalent to each other.

$$\forall\, x \quad \forall\, y \quad \exists\, z \qquad z - x \;=\; y \tag{10}$$

$$\forall\, x \quad \forall\, y \quad \exists\, z \qquad x > 0 \;\Rightarrow\; \frac{z}{x} \;=\; y \tag{11}$$

$$\forall\, x \quad \forall\, y \qquad x + y \;=\; y + x \tag{12}$$

$$\forall\, x \quad \forall\, y \quad \exists\, z \leq x + y \qquad z - x \;=\; y \tag{13}$$

Note that while Equation (10) can be transformed into a logically equivalent $\Pi_1^*$ sentence (encoded by Equation (13) ), there is no analogous $\Pi_1^*$ sentence equivalent to Equation (11). (This is because Addition but not Multiplication belongs to our set of U-Grounding functions.)

**Comment 1.6**. The logic literature uses similar ideas, but expresses them slightly differently than my notation in Definition 1.4. It begins with a language that recognizes Addition and Multiplication as its starting function symbols, and it builds all its terms out of these two function symbols rather than from my grounding language (which does not employ Multiplication as a function symbol). In such an arithmetic language, the traditional literature uses the terms $\Delta_0$ , $\Pi_i$ and $\Sigma_i$ formulae to have the identical definitions as I employ, except that its rules for building terms rely upon the Addition and Multiplication function primitives instead of my U-Grounding functions. The reason I resist using the traditional terminology will be explained in class.

**Lemma 1** *The set of axioms appearing in Table I are sufficient to prove for any integer $n$ the following theorem:*

$$\forall\, x \;\leq \overline{n} \quad [\; x = \overline{0} \;\vee\; x = \overline{1} \;\vee\; x = \overline{2} \;\vee\; \ldots \vee x = \overline{n} \;] \tag{14}$$

**Proof:** The proof is trivial and therefore omitted. The intuitive reason that you do not need to go through the proof is that if Table I's axiom set were not rich enough for the lemma's claim to be true, then I could trivially have added a couple more axioms to this table's list of axioms to make the lemma true. □

**Definition 1.7** Recall that a formula is called a "sentence" iff it contains no "free" variables. Similarly if a $\Delta_0^*$ formula contains no free variables then it is called a $\Delta_0^*$ sentence.

**Lemma 2** *Let $\Psi$ denote any $\Delta_0^*$ "sentence". Then the set of axioms appearing in Table I can prove either $\Psi$ or $\neg\,\Psi$ .*

**Proof:** I will only sketch the proof of Lemma 2 because it is trivial. Let us call a $\Delta_0^*$ sentence **degenerate** iff it contains no quantifiers. It is obvious that Lemma 2 is true for degenerate sentences because I had defined Table I's axioms so it can always determine the truth of any degenerate $\Delta_0^*$ sentence by an essentially routine construction.

On the other hand, every non-degenerate $\Delta_0^*$ sentence can have its truth verified by reducing its proof to the examination of a finite number cases, each being a proof of a degenerate $\Delta_0^*$ style theorem. This is because our proof methodology can use Lemma 1 to recognize that $\forall\, x \leq \overline{n}\;\; \Phi(x)$ is true if and only if each of the wffs of $\Phi(0)$, $\Phi(1)$ ... $\Phi(n)$ are provable, and likewise $\exists\, x \leq \overline{n}\;\; \Phi(x)$ is true if and only if at least one of $\Phi(0)$, $\Phi(1)$ ... $\Phi(n)$ are provable. Hence without delving into quite routine details, the idea is that each bounded quantifier of range say n will increase the proofs length by no more than a multiplicative factor proportional to approximately n's size. □

**Example 1.8.** It is easiest to appreciate the preceding proof by considering the example of the following $\Delta_0^*$ sentence:

$$\forall\, x < \overline{n} \quad \exists\, y < \overline{m} \quad \forall\, z < \overline{n} + \overline{m} \qquad \Psi(x,y,z) \tag{15}$$

One can essentially determine the truth of this sentence with a computer program that consists of three nested Do-Loops of sizes $n$ , $m$ and $n+m$ . In essence, such a 3-stage Do-Loop can be reduced to a proof that exhaustively examines the truth of each $\Delta_0^*$ sentence of the form $\Psi(\ \bar{i}\ ,\bar{j}\ ,\bar{k}\ )$ where $i\ <\ n$ , $j\ <\ m$ and $k\ <\ n+m$ , and then combines the results of these $n \cdot m \cdot (n+m)$ mini-proofs into a proof that determines whether Equation (15) or its negation is true.

**Comment 1.9** Let $L^-$ denote a language that differs from $L$ by

1. containing no Addition or Double function symbol
2. having one constant symbol $C_n$ to represent each integer $n$ .

The analogs of $\Delta_0^*$ , $\Pi_i^*$ and $\Sigma_i^*$ formulae in the language $L^-$ are called $\Delta_0^-$ , $\Pi_i^-$ and $\Sigma_i^-$ formulae. (In other words, these latter formulae are not allowed to use the growth functions of Addition or Doubling, and they recognize the existence of the integers 4,5,6..... by simply having an infinite number of constant symbols available.) It is easy to prove that an analog of Lemma 2 is true for $\Delta_0^-$ sentences.

**Definition 1.10** Let ADD$(x,y,z)$ and MULT$(x,y,z)$ denote two $\Delta_0^-$ formulae that are satisfied precisely when the two conditions of $x+y=z$ and $x*y=z$ are true. Some examples of such $\Delta_0^-$ definitions of ADD$(x,y,z)$ and MULT$(x,y,z)$ are provided by Equations (16) and (17) below

$$z - x = y \tag{16}$$

$$[\ (x=0 \vee y=0) \Rightarrow z=0\ ]\ \wedge\ [\ (x \neq 0 \wedge y \neq 0\ )\ \Rightarrow\ (\ \frac{z}{x} = y\ \wedge\ \frac{z-1}{x} < y\ \ )\ ] \tag{17}$$

We will say an axiom system $\alpha$ can recognize Addition **"as a total function"** iff it can prove

$$\forall\ x\ \ \forall\ y\ \ \exists\ z \quad \mathrm{Add}(x,y,z) \tag{18}$$

Similarly, we will say an axiom system $\alpha$ can recognize Multiplication **"as a total function"** iff it can prove

$$\forall\, x \;\; \forall\, y \;\; \exists\, z \quad \text{Mult}(x, y, z) \tag{19}$$

Also, we will say an axiom system $\alpha$ can recognize Successor (i.e. the operation of "plus one") as a total function iff it can prove

$$\forall\, x \quad \exists\, z \quad \text{Add}(x, 1, z) \tag{20}$$

Some axiom systems $\alpha$ are unable to prove Multiplication is a total function but can prove every true $\Delta_0^*$ sentence. Other axiom systems are unable to recognize any of Addition, Multiplication or Successor as total functions, but they can prove every valid $\Delta_0^-$ sentence. It will turn out these facts will be central to understanding the generality and limitations of Gödel's Second Incompleteness Theorem.

**Fact 1.11** A sentence $\Psi$ is said to be written in **Prenex Normal Form** iff for some $i \geq 0$ , it can be written as a $\Pi_i^*$ or a $\Sigma_i^*$ sentence. (Please see footnote [1] for an important clarifying comment.) It can be formally proven that a predicate logic using the language $L$ can always formally prove that for every sentence $\Phi$ there exists a prenex sentence $\Phi^*$ such that

$$\Phi \quad \Leftrightarrow \quad \Phi^* \tag{21}$$

The proof of this fact is not hard, but it is a bit tedious. It appears in Section 2.10 of Mendelson's book [3].

We will not use the observations of Paragraphs 1.9 through 1.11 in the next two sections of these Lecture Notes. However, I mention them because I am likely to briefly survey one of my recent papers on Self-Justification at the end of this course. They [7, 8, 9] will use the notation from these three paragraphs.

[1] Following Definition 1.4's notation convention, every $\Pi_i^*$ sentence can be rewritten as a $\Sigma_{i+1}^*$ sentence and viceversa. Therefore Fact 1.11's first sentence does not technically need to refer to both $\Pi_i^*$ and $\Sigma_i^*$ sentences in its definition of Prenex Normal Form.

## 2 The First Incompleteness Theorem

This section will sketch a proof of a version of the First Incompleteness Theorem. I will omit some details in the proof that are basically trivial to verify. Our version of the First Incompleteness Theorem will be less powerful than the stronger version in Mendelson‘s textbook [3]. However, its first virtue is that it is quite easy to prove. Also, our discussion will capture the main intuition behind the Incompleteness Theorem without going into excessive details.

Each formal proof in the Predicate Order Logic can be thought of as a string of characters, encoded as an integer. There are many methods for encoding such a proof. One sample method is illustrated in the Appendix A . It uses an alphabet of 55 characters for encoding a formula. If each character is represented by a symbol $D_i$ that corresponds to an integer between 0 and 54, then a sentence consisting of $n$ characters can be thought of as an integer $D_1, D_2, D_3, \ldots D_n$ , written as a number in base 64 whose value falls between $64^{n-1}$ and $64^n - 1$ . (See the Appendix A for one example of such an encoding method.)

The method for encoding a proof will be similar to a formula's encoding, except that it will need a 56-th symbol (called the period symbol) for separating the distinct formulae that appear in sequential order in the proof. As before, the sequence of $n$ digits $D_1, D_2, D_3, \ldots D_n$ , describing a proof, will again be written as a number in base 64, whose value falls between $64^{n-1}$ and $64^n - 1$ .

**Major Notation Convention:** The Logic literature uses the phrase **Gödel Number** or **Gödel Encoding** to describe the positive integer $N$ that represents the sequence of characters $D_1, D_2, D_3, \ldots D_n$ as a Base-64 integer. (Actually, the literature offers several different possible Gödel-like encoding schemes, and the one used in Section 3.4 of Mendelson's textbook [3] is much more complicated than the variant I define in Appendix A.)

**Definition 2.1.** An axiom system $\alpha$ will be said be a $\Delta_0^*$ axiom system if there exists a $\Delta_0^*$ formula $\mathrm{Ax}_\alpha(s)$ which yields a value of True for exactly those integers $s$ which correspond to Gödel encodings for one of $\alpha$'s axioms.

**Comment 2.2.** It turns out that all conventional axiom systems have $\Delta_0^*$ encodings. (This is because any axiom system that fails to be $\Delta_0^*$ encode-able is essentially impossible for a Human Being to interpret and contemplate !! )

**Lemma 3** *Assume that $\alpha$ is a $\Delta_0^*$ axiom system. Then it is possible to construct two $\Delta_0^*$ formula Logical$(x)$ and Prf$_\alpha(p,t)$ with the following two properties*

A. *Logical$(x)$ is true exactly when $x$ is the Gödel number of one of Mendeslon's Logical axioms.*

B. *Prf$_\alpha(p,t)$ is true exactly when $p$ is the Gödel number of a proof using $\alpha$'s proper axioms whose last sentence has $t$'s Gödel number. (Here $t$ is called the "theorem" proven by $p$'s list of sentences.)*

**Proof Sketch.** The method for proving Claims A and B is essentially the same type of methodology as would be employed to write a computerized language translator, similar to a compiler, assembler or interpreter. Interestingly, Gödel conceived of this method before any computer had been formally built.

We will not provide a formal proof of Claims A and B because the formal proofs are almost as lengthy as the writing of a computerized language translator — while at the same time they are conceptually as trivial and as routine as such a computerized object. □

**Lemma 4** *It is possible to encode a $\Delta_0^*$ formula, called Subst$(g,h)$, that yields a value of True precisely when $(g,h)$ satisfies the following condition:*

> The integer $g$ is an encoding of a formula and $h$ encodes a sentence identical to $g$, except that all free variables in $g$ are now replaced with the term $\overline{g}$ (defined by Comment 1.1's last sentence).

**Proof Sketch.** Once again, the $\Delta_0^*$ encoding for $\text{Subst}(g, h)$ is routine and omitted for the sake of brevity. If you are interested in going through the details, analogs of this encoding can be found in many logic textbooks. For instance, Items 9 and 10 on pages 194-195 in Mendelson's textbook are essentially analogous to our treatment of $\text{Subst}(g, h)$. (These items are not easy to read line-by-line because their careful reading requires a knowledge of perhaps two dozen pages that precede them.) $\square$

**Added Comment:** As with Lemma 3's proof, the proof of Lemma 4 is essentially analogous to the writing of a straightforward computer program (in this case using the language of the U-Grounding Functions and the $\Delta_0^*$ formulae to achieve the claimed functionality). It therefore should be trivially obvious to a computer science student that both these lemmas are true.

**Definition 2.3.** Let us recall that the deduction system, defined in Section 2 of Mendelson's textbook, is called a "Hilbert-style" proof. Let $\alpha$ again denote a $\Delta_0^*$ axiom system. We will now define a $\Pi_1^*$ sentence, called $\mho(\alpha)$ , whose translation into English amounts to the following sentence

> $*$ There is no Hilbert-style proof of **THIS SENTENCE** (looking at itself) from the axiom system $\alpha$ .

To achieve this task, we let $\Gamma(g)$ denote the following formula

$$\forall h \, \forall p \quad \{ \quad \text{Subst}(g, h) \quad \Rightarrow \quad \neg \, \text{Prf}_{\,\alpha} \, ( \, h \, , \, p \, ) \quad \} \tag{22}$$

Let $N$ denote the Gödel number of Equation (22)'s formula, and let us recall that the term $\overline{N}$ was defined by Comment 1.1. Then the $\Pi_1^*$ sentence that encodes $*$ (whose formal name is often written as " $\mho(\alpha)$ " ) is the sentence $\Gamma( \; \overline{N} \; )$ .

In other words from Equation (22), this implies that our $\Gamma( \; \overline{N} \; )$ sentence (that is often written as " $\mho(\alpha)$ " ) has the following encoding:

$$\forall h \, \forall p \quad \{ \quad \text{Subst}( \; \overline{N} \; , \; h \; ) \quad \Rightarrow \quad \neg \, \text{Prf}_{\,\alpha} \, ( \, h \, , \, p \, ) \quad \} \tag{23}$$

**Definition 2.4.** A $\Delta_0^*$ axiom system $\alpha$ will be called **Regular** iff it either includes all Table I's axioms or at least can prove all Table I's axioms as theorems. (The latter condition is as good as containing Table I's axioms because after proving Table I's 36 statements, $\alpha$ can use them as if they were axioms during the proof of any other theorem.)

**Lemma 5** *Suppose that $\alpha$ is a Regular $\Delta_0^*$ axiom system that is consistent. Then $\alpha$ must be unable to prove the sentence $\mho(\alpha)$ .*

**Proof:** For the sake of constructing a proof-by-contradiction, let us assume that Lemma 5 was false and therefore that $Q$ represented a proof of $\mho(\alpha)$ from the axiom system $\alpha$ . Let $K$ denote $\mho(\alpha)$ 's Gödel number. Then by definition, the statements in Equations (24) and (25) are both true.

$$\text{Subst}(\ \overline{N}\ ,\ \overline{K}\ ) \tag{24}$$

$$\text{Prf}_{\alpha}\ (\ \overline{K}\ ,\ \overline{Q}\ ) \tag{25}$$

Since Equations (24) and (25) represent logically valid $\Delta_0^*$ formulae, it follows from Lemma 2 that both these sentences are provable from $\alpha$ .

The next point is that it is trivial to verify that the conjunction of Equations (24) and (25) implies Equation (23) is false. Hence since $\alpha$ can prove (24) and (25), it must be able to prove $\neg\ \mho(\alpha)$ . (This is easy to prove as a homework exercise.)

Next, we recall that the opening paragraph of our proof-by-contradiction had entertained the temporary assumption that $\alpha$ could prove $\mho(\alpha)$ . Hence from the last sentence of the prior paragraph, we conclude that $\alpha$ would then be inconsistent, since it would then prove both $\mho(\alpha)$ and its negation $\neg\ \mho(\alpha)$ (via the prior paragraph's construction).

The latter observation will now enable our proof-by-contradiction to reach its desired end. This is because Lemma 5's hypothesis indicated that $\alpha$ was consistent, but the last paragraph contradicted this assumption of $\alpha$'s consistency.

Hence to avoid this contradiction, we are forced to conclude the temporary assumption in our proof's first sentence (that $\alpha$ could prove $\mho(\alpha)$ ) must be false. The negation of this temporary assumption demonstrates that Lemma 5 must be valid (because its negation is inherently contradictory). $\Box$

**Lemma 6** *Again, suppose that $\alpha$ is a Regular $\Delta_0^*$ axiom system that is consistent. Then the sentence $\mho(\alpha)$ must be true.*

**Proof:** Lemma 5 demonstrated that $\alpha$ is unable to prove $\mho(\alpha)$ . However $\mho(\alpha)$'s formal statement (given in statement $*$ ) amounts to saying "There is no proof of $\mho(\alpha)$ from the axiom system $\alpha$ ". Therefore, we are forced to conclude that $\mho(\alpha)$ is a formally true statement. $\Box$

**Theorem 1** (A weak version of Gödel's First Incompleteness Theorem) *Again, suppose that $\alpha$ is a Regular $\Delta_0^*$ axiom system that is consistent. Then there exists a $\Pi_1^*$ sentence that is true but not provable from $\alpha$ .*

**Proof:** From Lemmas 5 and 6, it follows that $\mho(\alpha)$ is a formally true $\Pi_1^*$ sentence that cannot be provable from $\alpha$. $\Box$

**Definition 2.5.** An axiom system $\alpha$ is called **Recursively Enumerable** iff there exists a computerized procedure $P$ that if allowed to run indefinitely will produce a list of all of the Gödel numbers of $\alpha$'s axioms and which will list no other Gödel number besides these. Using this notation, we can now give the statement of a stronger version of Gödel's First Incompleteness Theorem:

**Theorem 2** *Every consistent recursively enumerable axiom system $\alpha$ is unable to prove some logically valid $\Pi_1^*$ sentence. (Also, $\alpha$ cannot prove every logically valid $\Pi_1$ sentence.)*

The proof of Theorem 2 has a generally similar structure to Theorem 1's proof.

However, its details (which appear in Mendelson's textbook) are somewhat more complicated. They will not appear here.

**Comment 2.6.** Theorem 2 is stronger than Theorem 1 because it indicates that no computerized algorithm can be used to generate an axiom system that enables us to find all true $\Pi_1^*$ or $\Pi_1$ sentences ! It can be viewed as saying that Artificial Intelligence will never be able to reach the full goal that at least science fiction writers have set for it ! The theory of NP-hardness is also related to this issue.

**Comment 2.7.** In my view, the main philosophical implication of Gödel's Incompleteness Theorem is that computer science and computers can never achieve a set of over-ambitious goals, because of the "undecidability problems" raised by Gödel's Incompleteness Theorem. Rather a more mature approach for computer science is to restrict itself to well-defined narrowly constructed engineering-style problems — where the essentially "undecidable tasks" that are unrealistically over-ambitious are avoided.

The opening paragraph of the article about Gödel in the March 19, 1999 issue of Time Magazine [5] stated that "many scholars consider Gödel's Incompleteness Theorem to be the *most important result* of 20-th century mathematics". The editors of Time Magazine were very careful to use the word "many" in the preceding sentence because there is no unanimous opinion among scholars about what are the say five most important results in 20th century mathematics. However my point is that while some scholars may not agree with the cautious skepticism I expressed in Comment 2.7 about the importance for Computer Science focusing on narrowly defined problems, there should be no question that the Incompleteness Theorem is an important result, that has implications about what directions Computer Science can and cannot profitably pursue.

But what does the Incompleteness Theorem really mean? And what does it imply to for computer science when Theorem 2 applies to all "recursively enumerable" axiom systems?

The answer to these questions are likely to be quite challenging. They may not be resolved during our life times. Or ever?

## 3 The Second Incompleteness Theorem

The axiom system Peano Arithmetic (PA) was defined by Section 3.1 of Mendelson's textbook. It essentially has three function symbols corresponding to Addition, Multiplication and Successor and thus recognizes these three operation as "total" (using Definition 1.9's notation). Its chief feature is that it includes the Principle of Induction as an axiomatic assumption (i.e. see page 155's S-9 axiom scheme.)

**Definition 3.1**. Let $\alpha$ and $\beta$ denote two axiom systems. We will say $\beta$ is an **Extension** of $\alpha$ and write $\beta \supset \alpha$ iff $\beta$ can prove all $\alpha$ 's theorems.

In Gödel's 1931 paper [1], both his First and Second Incompleteness Theorems had appeared. The former states that neither Peano Arithmetic nor any recursively enumerable extension of it can prove all the true $\Pi_1$ sentences. The Second Incompleteness Theorem states that if $\alpha$ represents either Peano Arithmetic or any recursively enumerable extension of it, then $\alpha$ must be unable to prove its own consistency.

Generalizations of the Incompleteness Theorem, subsequent to 1931, have shown both the First and Second Incompleteness Theorems also apply to many axiom systems weaker than Peano Arithmetic (PA). The only author to develop a partial exception to the Second Incompleteness Theorem was Willard (for certain axiom systems weaker than PA).

We will not discuss my work in this handout. Our goal in this section will instead be to prove a version of the Second Incompleteness Theorem. My work will be discussed in the Fall Semester in detail and perhaps summarized in the last two weeks of this course as well.

**Definition 3.2**. The axiom system PA+ will be defined to be a **minor extension**

of Peano Arithmetic (PA) that contains function symbols for all the U-Grounding Functions, in addition to the function symbols for Addition and Multiplication. Thus, PA+ will include the nine axioms given in Section 3.1 of Mendelson's textbook plus the 36 axioms listed in Table I, defining the U-Grounding functions. You should note that:

1. PA+ recognizes the validity of the Principle of Induction because it includes the axiom scheme S-9 from page 155 of Mendelson's book.
2. The 36 axioms from Table-I cause PA+ to be an essentially trivial extension of PA, by allowing it to use the language of the U-Grounding function symbols.

**Fact 3.3** The key point in understanding Gödel's Second Incompleteness Theorem is that Lemma 5's proof can be formally done by the axiom system PA+ (indeed also by PA). This means that PA+ can prove the following sentence:

$**$ No consistent and regular $\Delta_0^*$ axiom system $\alpha$ can prove the sentence $\mho(\ \alpha\ )$.

The intuitive reason that PA+ can prove $**$ is that it contains the axioms of the Principle of Induction, and the latter is all that is needed for an axiom system to formalize the proof of Lemma 5.

**Definition 3.4**. The symbol Cons( $\alpha$ ) will denote a $\Pi_1^*$ sentence declaring the axiom system $\alpha$ is consistent.

**Lemma 7** *Let us assume the axiom system PA+ is consistent (because all of conventional mathematics would collapse if it was not). The axiom system PA+ then cannot prove the* $\Pi_1^*$ *theorem, Cons( PA+ ), declaring its own consistency.*

**Proof:** For the sake of of constructing a proof-by-contradiction, let us entertain the temporary assumption that PA+ could prove Cons( PA+ ). Recall that Fact 3.3

noted that PA+ had a capacity to prove statement $**$ . Since PA+ can also prove that it is regular, it can thus combine these facts to formally prove the statement that:

$\#$ PA+ cannot prove $\mho($ PA+ $)$ .

We next observe that the statement # can be proven by PA+ to be logically equivalent to the sentence $\mho($ PA+ $)$ . (This is simply because PA+ can trivially verify that the only integer $K$ satisfying Subst$(N, K)$ in Equations (23) through (25) is $\mho($ PA+ $)$'s Gödel number.)

Thus, the preceding paragraph has established that PA+ can **BOTH** prove # and prove that # is logically equivalent to the sentence $\mho($ PA+ $)$. This implies that PA+ also has a simultaneous capacity

1. to prove the sentence $\mho($ PA+ $)$ (since it is equivalent to # ).
2. and also to **disprove** the sentence $\mho($ PA+ $)$ . (The latter is because we can easily repeat the techniques from the prior section to establish that if $Q$ represents a proof of $\mho($ PA+ $)$ **from axiom system PA+**, then it must be the case that PA+ can use essential analogs of the Section 2's methodology to prove $\neg\, \mho($ PA+ $)$ in the present context. Since a student can find "diagonalization-style" proofs to be confusing, I have inserted an Appendix B into these lecture notes — which essentially repeats Section 2's techniques to explain line-by-line the exact structure of the proof of $\neg\, \mho($ PA+ $)$ . ) .

Hence using the above kind of "diagonalization paradox", we can conclude that if PA+ proved Cons( PA+ ), then it would be inconsistent because Items 1 and 2 (above) show that it would then prove both a sentence (i.e. $\mho($ PA+ $)$ ) and its negation (i.e. $\neg\, \mho($ PA+ $)$ ).

Since PA+ is presumed to be consistent by conventional mathematics, it follows that PA+ must be unable to prove its own consistency, to avoid this clearly impossible condition. $\square$

**Clarification 3.5**. Let me do the last part of the proof above in slow motion so that you understand, line-for-line, what it exactly intends to undertake. Suppose that you proved the following two sentences

1. $X \Rightarrow Y$

2. $\neg Y$

Then of course, you could conclude that $\neg X$ is true. The point is that the preceding proof of Lemma 7 uses the above reasoning where

A. X is the statement that "PA+ proves Cons( PA+ ) "

B. Y is the statement that PA+ is inconsistent.

Since we know PA+ is consistent, we know $\neg Y$ is true (as item 2 requires). Moreover, Item 1 is also true because the heart of Lemma 7's proof verified it. Hence, we conclude that $\neg X$ is true, which is exactly what Lemma 7 had claimed !!!

In other words, what I did in Paragraph 3.5 (above), was repeat in slow motion the last part of Lemma 7's proof, so that you can have a better understanding of its intuition !!!!

**Theorem 3** (A version of Gödel's Second Incompleteness Theorem) *Let $\alpha$ denote any $\Delta_0^*$ encode-able axiom system that is an extension of PA+ and which is consistent. The axiom system $\alpha$ then cannot prove the $\Pi_1^*$ theorem, Cons( $\alpha$ ), declaring its own consistency.*

**Proof:** By literally the identical reasoning as was used in Lemma 7's proof. In particular, the same reasoning that Lemma 7 used to prove PA+ could not prove Cons( PA+ ) also establishes that $\alpha$ cannot prove Cons( $\alpha$ ). $\square$

**Comment 3.6**. The original version of the Second Incompleteness Theorem by Gödel was slightly stronger than Theorem 3. It stated that no recursively enumerable axiom system that was an extension of PA (not PA+) could prove its own consistency. Also, let $Q^*$ denote an axiom system, identical to PA, except that the Principle of Induction is removed. (This means that $Q^*$ differs from PA's definition on Page 155 by removing all the S-9 axioms.) In 1985, a JSL paper by Pudlak [4] essentially showed that no extension of $Q^*$ could prove its own consistency.

**Comment 3.7**. We will discuss the Second Incompleteness Theorem in more detail during the second semester of this course. Our discussion will include the generalizations of it (by myself and others) and also partial boundary-case exceptions to it that I developed.

Although the Second Incompleteness Theorem is technically correct, there is a very deep philosophical question it raises — with broad implications for Computer Science and other disciplines. It is the following question

> Why do Human Beings choose to think and cogitate if they did not have faith that Thinking is a useful process? The problem is that Theorem 3 states that any formalism having faith in its own consistency will collapse in inconsistency. How do Human Beings manage to grapple with this inherent dilemma in their day-to-day life? How do Human Beings motivate themselves to gather the energy to think if they did not have faith in the consistency of their thought processes — notwithstanding the problems raised by Gödel's two theorems? And how will the computers of the future attempt to imitate the higher Human Thinking process when they confront the same problem?

My papers about Self-Justifying axiom systems, discussed next semester, illustrate how some non-conventional Logics can escape the reach of Gödel's Second (but not First) Incompleteness Theorem and possess some knowledge about their own consistency.

I don't want to over-state the importance of what I am doing because both Gödel's theorems are technically correct. On the other hand, the web site for the *Journal of Symbolic Logic* describes itself as "Logic's foremost journal", and my paper [7] was the longest of the 120+ papers published by the JSL during 2001. It described non-conventional logics that evade the Second Incompleteness Theorem. If you are interested, they will be discussed next semester.

## 4 Other Comments about Incomplteness and this Course

In addition to exploring Section 2.7 in detail during this course, I also plan to do briefly survey Section 2.8 through 2.10. That survey will include a description of the statements proven in these sections, but not their formal proofs.

For example, the main Proposition 2.28 of Section 2.9 will explain that there is no meaningful difference between the axiom systems that I call PA and PA+ in these lecture notes. If $L'$ is a language that only has function symbols for representing the operations of Addition, Multiplication and Successor then both these axiom systems will prove the same set of theorems about the language $L'$, by Proposition 2.28.

Some logic books would call PA+ a "conservative extension" of PA because it proves added theorems only when one examines language primitives not in PA's language (i.e. the sentences about the "U-Grounding Functions").

Proposition 2.28 implies such a result when one (after an essentially very minor change in notation) sets $K = PA$ and $K^{\#} = PA+$. And so you may wonder why I chose to use the axiom system PA+ rather than PA during these lecture notes when there is no meaningful difference between them when they use the language of PA ??

The answer to this question is solely pedagogic. The main theorems are easier to prove when one uses a broader language. This is because of a phenomena, similar to the Deduction Theorem occurs, where one can compress a proof sharply by taking certain shortcuts.

And so the results about PA+ turn out to be easier to prove than their analogs about PA, although the latter is only a conservative extension of the former.

On the other hand, there is a historic reason why logicians have chosen mostly to work in the framework of PA, rather than PA+. It is that PA is the historic language of the simplest branch of Mathematics, since its basic function primitives are Addition and Multiplication (and nothing more). Godel's 1931 version of the Incompleteness Theorem shocked the world of Mathematics because it showed that undecidability results arose in such a foundational branch of Mathematics.

It also answered an open question that was explicitly raised by the mathematician David Hilbert.

## 5 Baby Set Theory and the Completeness Theorem

Let $\Gamma$ be a set of axioms for some language $L$, and $\Psi$ be any sentence in $L$'s language. Godel's Completeness Theorem states essentially that if $\Psi$ is valid in every model for $\Gamma$'s axioms then $\Psi$ will have a proof from $\Gamma$ using the Hilbert-deduction method, appearing in Section 2.3 of Mendelson's textbook. It thus explains why Section 2.3's proof formalism is very natural (by showing that the Hilbert Deduction system proves exactly the full set of theorems that one would want to have formally proven).

It is not our intention in this Section to repeat the proof of the Completeness Theorem that Mendelson presents in Section 2.7 (because we do not know a better way to present it). However, my impression is that Section 2.7's proof will be easier to read if a student examines my discussion in this section first. This is because Section 2.7 presumes a knowledge of Baby Set Theory, only briefly surveyed in the book's preface. I believe that I can make Mendelson's Section 2.7 easier to read, if I first present some short 4-page introduction to Baby Set Theory that will help you understand Section 2.7.

In our discussion, letter such as $S$ and $T$, will denote sets. We will say $S$ has cardinality less-than-or-equal to $T$'s cardinality (and write $S \preceq T$ ) iff there exists a one-to-one function, called say $F$, from the set $S$ into $T$. We will say that $S$ and $T$ have the same cardinality (and write $S \cong T$ ) if the function $F$ is an "onto" function from $S$ to $T$. That means every member of $T$ has some member of $S$ mapping onto it under our function $F$.

Also, the symbol $S \prec T$ denotes the identity:

$$S \preceq T \ \wedge \ \neg\, S \cong T \tag{26}$$

Below are three well known facts about the cardinality operator proven somewhere in Chapter 4 of Mendelson's book:

$$\forall\, S \quad \forall\, T \qquad [\, S \preceq T \ \wedge \ T \preceq S\,] \ \Leftrightarrow \ S \cong T \tag{27}$$

$$\forall\, S \quad \forall\, T \qquad S \preceq T \ \vee \ T \preceq S \tag{28}$$

$$\forall\, S \quad \forall\, T \qquad S \cong T \quad \Leftrightarrow \quad T \cong S \tag{29}$$

Some of the results about the cardinality operator are a bit counter-intuitive. For example, let $I$ be set of all positive integers, and let $E$ be the set of positive even numbers. Then the following lemma holds:

**Lemma 8** $\quad I \ \cong \ E$

**Proof:** Consider a function $F$ that maps the integer $x$ onto $2x$. Since $F$ is a 1-to-1 and onto function, it implies $I \ \cong \ E$. $\square$

Let Power($S$) be defined as the set of all subsets of the set $S$. It is so-named because the power-set of a finite set of $n$ elements has a cardinality of exactly $2^n$ elements.

Also, define $S \times T$ to be the cross product of the sets $S$ and $T$. (It consists of all ordered pairs (x,y) where $x \in S$ and $y \in T$.) We will also use the symbol Square($S$) to denote the cross-product set $S \times S$.

For sets of finite cardinality, it is of course true that:

$$S \prec \text{Power}(S) \tag{30}$$

$$S \prec \text{Square}(S) \tag{31}$$

However, the generalization of the above two inequalities is more complicated for sets with infinite cardinalities. It turns out that Equation (30) holds for all sets $S$ of infinite cardinality. However, Equation (31) does not also hold. Among infinite-sized sets, Equation (31) should be replaced by:

$$S \cong \text{Square}(S) \tag{32}$$

For the sake of simplicity, I will prove the validity of Equations (30) and (32) for the case where $S$ is the set of positive integers denoted as $I$. It should be emphasized these identifies also hold for **ALL SETS** of infinite cardinality.

**Lemma 9** $I \prec \textit{Power}(I)$

**Proof:** Let $T_i$ denote an arbitrary subset of $I$. Define the characteristic sequence of $T_i$ to be an infinite sequence of Boolean values $b_i^1$ , $b_i^2$ , $b_i^3$ ,... where

1. $b_i^j = 1$ when $j \in T_i$
2. $b_i^j = 0$ when the above membership relation fails.

We will prove Lemma 9 by using a proof-by contradiction. Let us therefore temporarily entertain the hypothesis that the lemma was false. Then there would exist a 1-to-1 function from $I$ onto Power($I$). We will show that this is impossible by constructing a subset, called $T_*$ of $I$ , that does not have any integer $x \in I$ mapping onto it (under $F$ ).

Let us define the bit sequence $c^1$ , $c^2$ , $c^3$ , ... to have $c^j$ represent the Boolean value of $\neg\, b^j_j$ . Let us define $T_*$ to be the subset of $I$ whose characteristic sequence is $c^1$ , $c^2$ , $c^3$ , ... .

It is apparent that for each i that $T_i \neq T_*$ (because their two characteristic functions will differ in their i-th bit values). Hence, our proof-by-contradiction has accomplished its purpose because it demonstrated that it is impossible for any integer $i$ to have $F(i) = T_*$

□

**Comment 5.1.** The above proof was devised by Cantor during the 19-th century. It is the historically first example of what logicians call a "Diagonalization Proof". (All diagonalization proofs, similar to the Incompleteness proofs in the prior two sections, are proofs-by-contradiction. They thus always have a non-constructive flavor to them.)

**Lemma 10** $I \cong \mathit{Square}(I)$

**Proof:** Consider a function $F$ that has the following definition:

1. $F$ maps the integer $1$ onto the ordered pair $(1,1)$.
2. $F$ maps the integers $2$ and $3$ onto the ordered pairs of $(2,1)$ and $(1,2)$.
3. $F$ maps the integers $4$ , $5$ and $6$ onto the ordered pairs of $(3,1)$ , $(2,2)$ and $(1,3)$.
4. $F$ maps the integers $7$ , $8$ , $9$ and $10$ onto the ordered pairs of $(4,1)$ , $(3,2)$ , $(2,3)$ and $(1,4)$.
5. etc.

It is obvious that such a function $F$ establishes that $I \cong$ Square$(I)$ □

**Definition 5.2** The literature calls a set $S$ **Denumerable** or **Countable** iff it has the same cardinality as the set of positive integers. Thus, for example the sets $E$ and Square($I$) were shown to be "denumerable" (or "countable") by Lemmas 8 and 10, while Lemma 9 showed that Power($I$) was not also denumerable.

**Comment about Mendelson's Book**. It often uses the words " Denumerable" or "Countable" from Definition 5.2. However, the definitions of these constructs appears only in a very tersely written passage in the textbook's preface. That is the main reason why I felt it would be helpful to write this section. It should help you better understand Section 2.7's discussion.

**Definition 5.3** The symbol $\aleph_0$ has also been used to denote the cardinality of the set of positive integers. Under set theory notation, $\aleph_{i+1}$ is the cardinality of a set whose size is equivalent to the power-set of a set of $\aleph_i$ cardinality.

**How This Section Should Have Helped You:** Each of the concepts defined above appear in Section 2.7 of Mendelson's textbook, and that is why I thought it might be helpful for my lecture notes to have a short 4-page section reviewing these definitions. I am not sure whether or not I will cover Corollary 2.20 in this course. I will definitely not do the parts of Section 2.7 that come after it. (You should read at least Corollary 2.20's statement — although perhaps not its proof.)

## Appendix A : Summary of Gödel Encoding Method

This appendix will briefly summarize our formal method for generating Gödel numbers for logical sentence and proofs. This encoding scheme will be roughly analogous to the natural B-adic encoding methods described by Hájek-Pudlák [2] and Wilkie-Paris [6] — insofar as the number of utilized bits to encode a semantic object will be approximately proportional to the length of such an expression written by hand.

Our encoding scheme will use the following 24 language symbols to formalize a logical sentence or proof:

1. The standard connective symbols of $\wedge$, $\vee$, $\neg$, $\rightarrow$, $\forall$ and $\exists$.
2. The left and right parenthesis symbols, and also a comma symbol (to separate the input arguments for a function).
3. Ten function symbols for representing the ten U-grounding functions of subtraction, division, logarithm, etc.
4. The relation symbols of " $=$ " and " $\leq$ ".
5. The symbol $\hat{V}$ for designating the presence of a basic variable symbol.
6. The symbol $\hat{C}$ for designating the presence of a constant symbol $C$ .
7. A special period symbol for separating the formulae that appear within a proof

Let a byte denote an unit of six bits. Define a proof as being either a sequence of bytes (or equivalently being an integer, written in base 64). Each of the 24 symbols (above) will be given some unique 6-bit code, ranging between 32 and 55. Our method for representing the presence of the i-th variable $v_i$ will be to encode it is as a string of $\lceil log_{32}(i+1) \rceil + 1$ bytes, where the first byte is the "$\hat{V}$" symbol and the remaining bytes encode i as a base-32 number. The same convention will be used to denote the presence of the i-th constant symbol $C_i$ except its first byte will be the "$\hat{C}$" symbol.

Our byte-styled encoding method will produce proof strings whose length is approximately proportional to the effort write down such a proof by hand. The Logic literature uses the phrase **Gödel Number** or **Gödel Encoding** to describe the positive integer $N$ that represents the sequence of characters $D_1,\ D_2,\ D_3,\ ...\ D_n$ as a Base-64 integer. (Actually, the literature offers several different possible Gödel-like encoding schemes, and the one used in Section 3.4 of Mendelson's textbook [3] is more complicated than the variant defined in this Appendix.)

There are many analogs of such types of encodings in the prior literature (see for example the Hájek-Pudlák textbook [2] ). Such compressed encodings are usually considered to be preferable and more efficient than an uncompressed encoding method, using say the Chinese Remainder Theorem in Mendelson's textbook [3]. All our theorems have analogs under such uncompressed encoding methods, but they are substantially more meaningful when one uses efficiently compressed encodings.

## Appendix B : How to Construct the Proof of $\neg\ \mho(\ \text{PA+}\ )$

This appendix will explain line-by-line how if $Q$ was a proof of $\mho(\ \text{PA+}\ )$ then we could also expand $Q$ into a longer proof of the opposing sentence $\neg\ \mho(\ \text{PA+}\ )$ . My proof is similar to techniques employed in Section 2. However, I feel no guilt repeating those techniques here because students may likely be feeling confused during the first few times that they witness diagonalization proofs. Essentially, the proof in this appendix is similar to the earlier discussion, except that I will be replacing the phrase $\mho(\ \alpha\ )$ with $\mho(\ \text{PA+}\ )$ .

To achieve our task, we need to first encode the sentence $\mho(\ \text{PA+}\ )$ . Let us recall how Section 2 had done this. We will again let $\Gamma(g)$ denote the following formula

$$\forall h\ \forall p \quad \{ \quad \text{Subst}(g,h) \quad \Rightarrow \quad \neg\ \text{Prf}_{PA+}\ (\ h\ ,\ p\ ) \quad \} \tag{33}$$

Let $N$ denote the Gödel number of Equation (33)'s formula, and let us recall that the term $\overline{N}$ was defined by Comment 1.1. Then as Section 2 had explained, the $\Pi_1^*$ sentence that encoded $\mho(\ \text{PA+}\ )$ was the sentence $\Gamma(\ \overline{N}\ )$ .

In other words from Equation (33), this implies that $\mho(PA+)$ (which is the same as $\Gamma(\ \overline{N}\ )$ ) is the following sentence:

$$\forall h\ \forall p \quad \{ \quad \text{Subst}(\ \overline{N}\ ,\ h\ ) \quad \Rightarrow \quad \neg\ \text{Prf}_{\,PA+}\ (\ h\ ,\ p\ ) \quad \} \tag{34}$$

Let $K$ denote $\mho(PA+)$ 's Gödel number. Then since $Q$ is assumed to be a proof of $\mho(PA+)$ , the statements in Equations (35) and (36) are both true.

$$\text{Subst}(\ \overline{N}\ ,\ \overline{K}\ ) \tag{35}$$

$$\text{Prf}_{\,PA+}\ (\ \overline{K}\ ,\ \overline{Q}\ ) \tag{36}$$

Since Equations (35) and (36) represent logically valid $\Delta_0^*$ formulae, it follows from Lemma 2 that both these sentences are provable from $PA+$ .

The final point is that it is trivial to verify that the conjunction of Equations (35) and (36) implies Equation (34) is false. Hence since $PA+$ can prove (35) and (36), it must be able to prove $\neg\ \mho(PA+)$ , because $\neg\ \mho(PA+)$ 's proof is constructed from the proofs of (35) and (36) via an entirely routine expansion of these other two proofs.

**Homework Problem:** Suppose that $p_1$ and $p_2$ are proofs of the $\Delta_0^*$ sentences in Equations (35) and (36). Finish the last paragraph of the above proof by showing how to construct from $p_1$ and $p_2$ a third proof, called say $p_3$ , of the sentence $\neg\ \mho(PA+)$ .

**Added Remark.** I will repeat a comment that was made on Page 1. You are the **first class** to receive these lecture notes. Therefore, I ask you to let me know what typos you have found and which points confuse you? For example, if I did not carefully define some construct that you wish to understand, please let me know what it is ?

These lecture notes will be improved by your feedback.

## Table I: List of $\Pi_1$ Axioms Defining The U-Grounding Functions and The Relation Predicates of $=$ , $\leq$ and $<$

1. $\forall x \, \forall y \quad x < y \Leftrightarrow y > x$
2. $\forall x \, \forall y \quad x \leq y \Leftrightarrow y \geq x$
3. $\forall x \, \forall y \quad x \leq y \quad \Leftrightarrow [\, x = y \vee x \leq y - 1 \,]$
4. $\forall x \, \forall y \, \forall z \quad x + y = z \quad \Leftrightarrow \quad [\; z - x = y \wedge z \geq x \;]$
5. $\forall x \quad x + x = \text{Double}(x)$
6. $\forall x \;\; x = x$
7. $\forall x \, \forall y \;\; x = y \Rightarrow y = x$
8. $\forall x \, \forall y \, \forall z \;\; \{\, x = y \wedge y = z \,\} \Rightarrow x = z$
9. $\forall x \, \forall y \quad x = y \quad \Rightarrow \quad \text{Predecessor}(x) = \text{Predecessor}(y) \wedge \text{Logarithm}(x) = \text{Logarithm}(y)$
10. $\forall x \, \forall y \, \forall a \, \forall b \quad \{\, x = a \wedge y = b \,\} \quad \Rightarrow \quad x - y = a - b \wedge \frac{x}{y} = \frac{a}{b} \wedge$

    $\text{Count}(x, y) = \text{Count}(a, b) \wedge \text{Maximum}(x, y) = \text{Maximum}(a, b) \wedge \text{Root}(x, y) = \text{Root}(a, b)$
11. $\forall x \, \forall y \, \forall a \, \forall b \quad \{\; x - y = a - b \wedge y = b \,\} \quad \Rightarrow \quad x = a$
12. $\forall x \, \forall y \, \forall a \, \forall b \quad \{\, x = a \wedge y = b \wedge x < y \,\} \quad \Rightarrow \quad a < b$
13. $\forall x \, \forall y \quad x = y \vee x < y \vee y < x$
14. $\forall x \, \forall y \, \forall z \;\; \{\, x < y \wedge y < z \,\} \Rightarrow x < z$
15. $\forall x \quad \neg\, x < x$
16. $\forall x \quad x < 1 \Rightarrow x = 0$
17. Predecessor(0)=0
18. $\forall x \quad x \neq 0 \quad \Rightarrow \quad \text{Predecessor}(x) < x$
19. $\forall x \, \forall y \;\; \neg\, [\, \text{Predecessor}(x) < y < x \,]$
20. $\forall x \, \forall y \, \exists z \leq y \quad x < y \Rightarrow x = \text{Predecessor}(z)$
21. $\forall x \quad x - 0 = x$
22. $\forall x \, \forall y \;\; y \neq 0 \Rightarrow x - y = \text{Predecessor}(\, x - \text{Predecessor}(y) \,)$
23. $\forall x \, \forall y \quad x < y \Rightarrow \frac{x}{y} = 0$
24. $\forall x \quad \frac{x}{0} = \frac{x}{1} = x$

25. $\forall x \; \forall y \quad x \geq y \geq 1 \;\Rightarrow\; [\; \frac{x}{y} > 0 \;\wedge\; \frac{x}{y} - 1 \;=\; \frac{x-y}{y} \;]$

26. $\forall x \; \forall y \;\; \{\; x \geq y \Rightarrow \text{Maximum}(x,y) = x \;\} \;\wedge\; \{\; y \geq x \;\Rightarrow \text{Maximum}(x,y) = y \;\}$

27. $\text{Logarithm}(0){=}0 \wedge \text{Logarithm}(1){=}1 \wedge \text{Logarithm}(2){=}2$

28. $\forall x \quad x \geq 3 \;\Rightarrow\; \text{Logarithm}(x) - 1 \;=\; \text{Logarithm}(\,\frac{x}{2}\,)$

29. $\forall x \quad \text{Count}(x,0) \;= \text{Count}(0,x) \;=\; 0$

30. $\forall x \quad x > 0 \quad\Rightarrow\quad \text{Count}(x-1,1) \;\neq \text{Count}(x,1) \;\leq\; 1$

31. $\forall x \; \forall y \quad \{\; \text{Count}(x,1) > 0 \vee \text{Count}(\frac{x}{2}, y-1) > 0 \;\} \quad\Rightarrow\quad \text{Count}(x,y) > 0$

32. $\forall x \; \forall y \quad \text{Count}(x,y) {-} \text{Count}(x,1) \;= \text{Count}(\frac{x}{2}, y-1)$

33. $\forall x \quad \text{Root}(0,x) = 0 \;\wedge\; \text{Root}(x,0) \;= \text{Root}(x,1) \;=\; x$

34. $\forall x \; \forall y \geq 2 \quad \text{Root}(x,y) \;\leq\; \text{Root}(\,\frac{x}{\text{Root}(x,y)}\,,\, y-1\,)$

35. $\forall x \; \forall y \geq 2 \; \forall z \quad z > \text{Root}(x,y) \quad\Rightarrow\quad z \;> \text{Root}(\frac{x}{z}, y-1)$

36. $\forall x \; \forall i \quad \text{Bit}(x,i) \;=\; \text{Count}(x,i)$ - $\text{Count}(x,i-1)$

It is essentially unimportant, but I hope I did not leave out any axioms. I did not have the axioms 1-5 or 36 in my JSL 2001 article because it used a different notation.

## References

[1] K. Gödel, " Über formal unentscheidbare Sätse der Principia Mathematica und Verwandte Systeme I", *Monatshefte für Math. Phys.* 37 (1931) pp. 349-360.

[2] P. Hájek and P. Pudlák, *Metamathematics of First Order Arith,* Springer 1991.

[3] E. Mendelson, *Introduction to Mathematical Logic*, Chapman & Hall (1997).

[4] P. Pudlák, "Cuts, Consistency Statements and Interpretations", *Journal of Symbolic Logic* 50 (1985) pp.423-442.

[5] March 29, 1999 issue of Time Magazine, pages 132-134.

[6] A. J. Wilkie and J. B. Paris, "On the Scheme of Induction for Bounded Arithmetic", *Annals of Pure and Applied Logic* (35) 1987, 261-302

[7] D. Willard, "Self-Verifying Systems, the Incompleteness Theorem and the Tangibility Reflection Principle", *Journal Symbolic Logic* 66 (2001) 536-596.

[8] D. Willard, "How to Extend The Semantic Tableaux And Cut-Free Versions of the Second Incompleteness Theorem Almost to Robinson's Arithmetic Q", *Journal of Symbolic Logic* 67 (2002) pp. 465-496.

[9] D. Willard, "Some New Exceptions for the Semantic Tableaux Version of the Second Incompleteness Theorem", in *Automated Reasoning with Semantic Tableaux and Related Methods*, Springer–Verlag LNAI#2381, (2002) 281-297.

## Appendix C: A Simpler Proof of Löb's Theorem

This appendix will provide a brief outline of a simpler proof for Löb's Theorem than appears in Mendelson's textbook.

I saw a published version of this proof. For the sake of simplifying its notation, it had assumed the germane axiom system was Peano Arithmetic and that the germane deduction method was the Hilbert-style formalism (appearing in Mendelson's textbook).

I am almost 100 % certain this technique could generalize for the paradigm appearing in Mendelson's textbook. However for simplicity, I will stick with the assumptions made by prior authors.

We will use the following notation:

1. $\Box\ (\ \Phi\ )$ will mean that there is a Hilbert-style proof of $\Phi$ from Peano Arithmetic (PA).
2. $\mathrm{PA}^{\neg\Psi}$ will denote the union of the axiom system PA with $\neg\ \Psi$ , in a context where $\Psi$ is *some special fixed axiom* that will be defined later.
3. $\Box^{\neg\ \Psi}\ (\ \Phi\ )$ will mean that there is a Hilbert-style proof of $\Phi$ from $\mathrm{PA}^{\neg\Psi}$ .

Obviously in general, $\Box^{\neg\ \Psi}\ (\ \Phi\ )$ and $\Box\ (\ \Phi\ )$ **WILL FAIL** to be equivalent to each other for most sentences $\Phi$ . However, there is one almost-silly-looking exception to this rule. That funny-looking exception leads to a much simpler proof of Löb's Theorem than had appeared in his 1955 paper (or most subsequent textbooks). It is desrcibed on the next two pages. It employs the special variants of $\Box^{\neg\ \Psi}\ (\ \Phi\ )$ and $\Box\ (\ \Phi\ )$ where the $\Psi$ (used in $\Box^{\neg\ \Psi}$ 's superscript) formally replaces $\Phi$ :

**Lemma 11** *The statements* $\Box^{\neg\Psi}\ (\ \Psi\ )$ *and* $\Box\ (\ \Psi\ )$ *are logically equivalent to each other.* (This strange-looking result holds roughly because " $\Box^{\neg\Psi}$ " was defined to mean that the *special sentence* " $\neg\ \Psi$ " is an additional axiom permitted in a proof.)

**Proof.** Almost Trivial. It is well known that $\Psi$ has a proof from PA if and only if it has a proof from PA that begins with the temporary assumption of $\neg\ \Psi$ and ends with the contradictory statement of $\Psi$ (itself). I forgot the theorem numbers, but this result has appeared in both the Enderton and Mendelson books.

The point is that by definition " $\Box^{\neg\Psi}\ (\ \Psi\ )$" is the contradiction proof mentioned in the preceding paragraph. Hence, it trivially follows that the statements $\Box^{\neg\Psi}\ (\ \Psi\ )$ and $\Box\ (\ \Psi\ )$ are logically equivalent to each other.

**Lemma 12** *The statement of Lemma 11 can be proven by Peano Arithmetic itself.*

**Proof.** Entirely routine generalization of Lemma 11's proof and omitted. (This "routine generalization" is messy to formalize, but trivial to intuitively appreciate.)

**Theorem 4** *(Löb's Seminal Theorem) If PA proves* $\Box\ (\ \Psi\ )\ \Rightarrow\ \Psi$ *, then PA will also prove* $\Psi$ *(by itself).*

**Proof by Contradiction:** Suppose for the sake of establishing a contradiction that PA proved Equation (37), BUT IT DID NOT ALSO PROVE $\Psi$

$$\Box\ (\ \Psi\ )\ \Rightarrow\ \Psi \tag{37}$$

Then $\mathrm{PA}^{\neg\Psi}$ must be consistent (since otherwise PA could prove $\Psi$ via a proof-by-contradiction).

However, it turns out we can show that Godel's Second Incompleteness Theorem will also imply that the prior sentence is false. This fact will enable us to prove Löb's Theorem (via a proof-by-contradiction) because the last sentence in the previous paragraph CANNOT be simultaneously true and false !

More precisely, PA$^{\neg\Psi}$ must be able to prove Equation (37) because PA could do this (and because PA$^{\neg\Psi}$ is an extension of PA). Also by definition PA$^{\neg\Psi}$ contains $\neg\ \Psi$ as an axiom. Hence, it can combine the latter fact with Equation (37) to derive

$$\neg\Box\ (\ \Psi\ ) \tag{38}$$

Furthermore since PA (and therefore also PA$^{\neg\Psi}$) can prove Lemma 11, it follows that PA$^{\neg\Psi}$ can use its knowledge of Equation (38)'s truth that to derive that Equation (39) is also true.

$$\neg\Box^{\neg\ \Psi}\ (\ \Psi\ ) \tag{39}$$

However, we will now deliver the punchline that will prove Löb's Theorm.

It is that from the definiton of " $\Box^{\neg\ \Psi}$ ", it must be ture that PA$^{\neg\Psi}$ can prove (39) ONLY IF IT HAS KNOWLEDGE of its own consistency. But Godel's Second Theorem specifies no consistent extension of PA can recognize its own consistency. Hence PA$^{\neg\Psi}$ *must consequently be* inconsistent.

The latter statement brings our proof-by-contradiction to its sought-after awkward end. This is because the last sentence of the first paragraph of Theorem 4's proof indicated that PA$^{\neg\Psi}$ was consistent, and the prior paragraph contradicted that assumption !

Hence, we have proven Theorem 4 because we have shown that the temporary assumption that it was false has led to a contradiction.

DECIDED NOT TO GIVE THESE QUESTIONS IN 2010

**Two Thinking Exercises that You Migh Wish to Try:** A Henkin Sentence is a sentence (defined with the Fixed Point Theorem) that says "There IS a proof of me from Peano Arithmetic." Note that the Henkin Sentences do differ from Godel Sentences by replacing the phrase "IS NOT" with "IS". Here are two thinking questions:

1. It takes only a couple of senteces to explain how one may use Löb's Theorem to verify PA can prove the Henking Sentence. What does this short tiny explanation look like?
2. Now here is a harder problem that may take you a day or more to do. *Without using Item 1's result,* prove that PA proves the Henkin sentence.

The second problem is very hard because it took mathematicians 20 years to answer Henkin's open question about whether or not PA proved the Henkin sentence. It is feasible for you to answer this question because it involves unpaking my current proof so that it proves the Henkin Sentence from first princibles. However, it took me a weekend to do this seeming straightforward task, and it will also take you some time *if you choose to try it ?*