Gödel's Incompleteness Theorem, A Critical Examination

Revision 2005/11/17 by Donald Wayne Stoner

Supporting material primarily from: Gödel, Escher, Bach: an Eternal Golden Braid, (G.E.B.) by Douglas R. Hofstadter, Random House, N.Y., C.1980. Also from Gödel's Proof, revised edition, (G.P.) by Ernest Nagel and James R. Newman.

Abstract:

It appears to me that the "proof" for Gödel's incompleteness theorem (almost universally regarded to be established truth) contains a critical error. This paper will present a version of that proof and will explain the error in enough detail that others can evaluate whether the mistake is Gödel's or mine.

Gödel's Paper:

Back in 1931, Kurt Gödel published his paper, "On Formally Undecided Propositions in Principia Mathematica and Related Systems I." It states, in its Proposition VI:

To every ω-consistent recursive class *κ of formulae* there correspond recursive *class-signs r,* such that neither *v* Gen *r* nor Neg (*v* Gen *r*) belongs to Flg(*κ*) (where *v* is the free variable of *r*).

This statement, like the rest of Gödel's paper, uses very technical terminology which I, personally, find extremely difficult to follow. Fortunately, books like G.E.B. (Hofstadter) and G.P. (Nagel and Newman), have made the necessary concepts accessible to people like me. This document is based on the understanding I was able to obtain from those two books - and not from Gödel's paper itself. Gödel's claim (paraphrased for us by Hofstadter) is that:

"All consistent axiomatic formulations of number theory include undecidable propositions." (G.E.B. p.17)

We can go a little bit farther into layman's territory by saying that a formal system of number theory must either be inconsistent (make inconsistent decisions about whether or not some statements are true) or be incomplete (be unable to demonstrate the truth of every true statement).

Formal Systems:

Gödel's Incompleteness Theorem applies to all formal systems of logic and mathematics. We might need to start by reviewing what a formal system is. The goal of a formal system in mathematics or logic is to sort true statements from false ones, using only rules and axioms which are agreed upon. These systems start with a core of axioms, and add theorems to these axioms by using a set of formal rules.

First, we will design a simple formal system which models some truth about addition. We will take the symbol "=" to be our only axiom and will use just two formal rules to generate additional theorems:

Rule 1:
A new theorem can be produced from an old one by adding an "I" to both ends. If "X" is a theorem, then so is "IXI".

Rule 2:
A new theorem can be produced by adding a "+" to the left side. If "X" is a theorem, then so is "+X".

By playing around with these rules, we can see that. I+I=II, I+II+III=IIIIII, and +I=I, are all theorems; but that II+II=I, =I=, and II=I+I, are not. The theorems can all be interpreted as true statements about addition (by interpreting "II" as 2, "III" as 3 etc.), but there are other true statements about arithmetic which cannot be formally generated using this system. Notice that II=I+I can be understood as a true statement, although it is not a "theorem" by the two rules we are using. We could add a third rule:

Rule 3:
A new theorem can be produced by adding a "+" to the *right* side. If "X" is a theorem, then so is "X+".

With this third rule, II=I+I now becomes a theorem of our formal system. However, there are still true statements about arithmetic which cannot be formally generated using this improved three-rule system.

This was just a very simple formal system which was made up to illustrate the difference between theorems and actual truth. Next, we will examine some of the real formal systems which are normally used to sort logical and mathematical truth from error.

Propositional Calculus: (P.C.)

(See G.E.B. p. 187 or G.P. pp. 45-56 for a similar system).

Rules of P.C.:

Joining Rule:
If p and q are both theorems then so is <p ∧ q>

Separation Rule:
If <p ∧ q> is a theorem then so are p and q

Double Tilde Rule:
~~ can be deleted from any theorem. It can also be inserted, provided that the result is well formed.

Fantasy Rule:
If q can be derived when p is assumed, then <p ⊃ q> is a theorem.

Carry-over Rule:
Inside a fantasy, any theorem from the "reality" one level up can be imported and used.

Rule of Detachment:
If p and <p ⊃ q> are both theorems, then q is a theorem.

Contrapositive Rule:
<p ⊃ q> and <~q ⊃ ~p> are interchangeable.

De Morgan's Rule:
<~p ∧ ~q> and ~<p ∨ q> are interchangeable.

Switcheroo Rule: (see G.E.B. p.187 for an "explanation" for the name)
<p ∨ q> and <~p ⊃ q> are interchangeable.

Here, the symbol "∧" means "and," "∨" means "or," "~" means "not," and "⊃" means "implies that." Triangular brackets, "<" and ">", are used to specify order of operation; and square brackets, "[" and "]", are used to mark off a "fantasy." Under P.C., all axioms must be generated using the fantasy rule.

Rule of Excluded Middle:

An important consequence of the P.C. is the Rule of Excluded Middle. A proposition must be either true or false. $\langle p \lor \sim p\rangle$ (It cannot be neither true nor false.) Formally, this is a normal "or" not an "exclusive or." This rule does not directly assert that a proposition cannot be both true and false $\sim\langle\sim p \land p\rangle$. However, it is a simple matter to demonstrate that this also is logically implied by the formal form of the rule:

(1)	$\langle p \lor \sim p\rangle$	
(2)	$\sim\sim\langle p \lor \sim p \rangle$	Double Tilde Rule (added) (1)
(3)	$\sim\langle\sim p \land \sim\sim p\rangle$	De Morgan's Rule (2)
(4)	$\sim\langle\sim p \land p\rangle$	Double Tilde Rule (removed) (3)

It turns out that the Rule of Excluded Middle is a very important one. If we allow that a proposition and its negation are both true, we can prove absolutely anything to be true:

	[			Begin Primary Fantasy
(5)		$\langle\sim p \land p\rangle$		Presumed for Fantasy
(6)		$\sim p$		Separation Rule (5)
(7)		p		Separation Rule (5)
		[		Begin Secondary Fantasy
(8)			$\sim q$	Presumed for Fantasy
(9)			p	Imported (Carry-over Rule) (7)
(10)			$\sim\sim p$	Double Tilde Rule (9)
		]		End Secondary Fantasy
(11)		$\langle\sim q \supset \sim\sim p\rangle$		Conclusion from Fantasy (8, 10)
(12)		$\langle\sim p \supset q\rangle$		Contrapositive Rule (11)
(13)		q		Detachment (6, 12)
	]			End Primary Fantasy
(14)	$\langle\langle\sim p \land p\rangle \supset q\rangle$			Conclusion from Fantasy (5, 13)

It doesn't matter what q (assumed to be false in line 8) means. It might mean, for example, "I am the Pope." Neither does it matter what p means. The point is, if both p and $\sim p$ are presumed to be true, then we know that q (line 13) must also be true. Absolutely anything can be concluded.

Self Stultification:

A self stultifying statement is one which supplies the information for its own refutation. Consider the statement:

"No statements are true."

This statement (which we will call statement q) asserts that there are no statements which are true. Yet q is itself a statement and must, therefore be false by its own assertion. Symbolically this could be expressed:

$\langle q \supset \sim q\rangle$

From this we can see:

(15) <q⊃~q>	
(16) ~~<~~q⊃~q>	Double Tilde Rule (15)
(17) ~~<~q∨~q>	Switcheroo Rule (16)
(18) ~<~~q∧~~q>	De Morgan's Rule (17)
(19) ~<q∧q>	Double Tilde Rule (18)
(20) ~q	Separation Rule (19)

This means statement q is a false statement. From this, we conclude that there must be at least one true statement. For example, we conclude from line 20 that the negation of q is true:

"It is not true that no statements are true."

We must remember that lines 15-20 really comprise a fantasy (since we don't know at the beginning whether q is true or false). The fantasy rule yields the true theorem:

(21) <<q⊃~q>⊃~q>

So we know the consequence of assuming q. It is that ~q must be true. By the rule of excluded middle, the only thing, other than q, we might have assumed must be ~q (making ~q true anyway). So either way, we know that ~q must be a true statement.

Epimenides Paradox:

In about 600 B.C., there was a poet/philosopher named Epimenides who made the statement, "All Cretans are liars!" Since Epimenides was, himself, from the island of Crete (and a poet), this statement has been understood to mean, "I'm a liar," or even, "This statement is false." Here we will word the Epimenides paradox as:

(22) "This statement is not true."

This statement (which we will call statement p) is similar to the self stultifying statement q, "No statements are true," except that it only applies to itself. The consequence of this restriction is very interesting and is also quite poetic. Like q, p asserts its own negation:

<p⊃~p>

But notice what happens if we assume that p is false. When we assumed q was false, it only meant that there was at least one statement which was true (not necessarily q itself). But when we assume p is false then it must be p itself which is now asserted (by negation) to be true. The Epimenides paradox is self stultifying in both directions:

<p⊃~p>
<~p⊃p>

Whatever we assume, we find the "truth" shifting on us. This statement has a circular or even oscillating nature. (The word "paradox" is defined differently in different fields of study. In this paper we will be using the term "paradox" in this circular sense implied by "Epimedies paradox.") If we take the Epimenides paradox seriously, we come to an interesting conclusion:

(23) <p⊃~p>	This Statement (p) is Not True
(24) <~p⊃p>	
(25) ~~<~~p⊃~p>	Double Tilde Rule (23)
(26) ~~<~p∨~p>	Switcheroo Rule (25)
(27) ~<~~p∧~~p>	De Morgan's Rule (26)
(28) ~<p∧p>	Double Tilde Rule (27)
(29) ~p	Separation Rule (28)
(30) <p∨p>	Switcheroo Rule (24)
(31) ~~<p∨p>	Double Tilde Rule (30)
(32) ~<~p∧~p>	De Morgan's Rule (31)
(33) ~~p	Separation Rule (32)
(34) p	Double Tilde Rule (33)
(35) <~p∧p>	Joining Rule (29, 34)

Although the preceding derivation gives us what we need (in line 35) to prove that we are all the Pope (lines 5-14), this does not make that conclusion valid. No one (not even Epimenides himself) expects us to take this seriously. The whole derivation (lines 23-35) ought to have been enclosed in fantasy brackets "[" and "]" making the real conclusion:

(36) < <<p⊃~p> ∧ <~p⊃p>> ⊃ <~p∧p> > Fantasy Rule (23, 24, 35)

So symbolic logic didn't really come crashing to the ground when Epimenides called himself a liar. We merely forgot to enclose our reasoning in fantasy brackets as we ought to have done.

Typographical Number Theory: (T.N.T.)

(see G.E.B. pp.216-218, 224 or similar information from G.P. pp. 70+)

The P.C. seems to be useful as far as it goes, but the T.N.T. extensions will take us farther.

Axioms of T.N.T.:
1) ∀a: ~Sa = 0
2) ∀a: (a + 0) = a
3) ∀a:∀b: (a + Sb) = S(a + b)
4) ∀a: (a · 0) = 0
5) ∀a:∀b: (a · Sb) = ((a · b) + a)

Rules of T.N.T.:

Rule of specification:
If variable u occurs in string x, and if ∀u:x is a theorem, then x is a theorem, and so is any string made by replacing u, wherever it occurs, by one and the same term.

Rule of Generalization:
If x is a theorem containing free variable u, then ∀u:x is a theorem. (Restriction: No generalization is allowed in a fantasy on any variable which appeared free in the fantasy's premise.)

Rule of Interchange:
If u is a variable, then the strings: $\forall u{:}\sim$ and $\sim\exists u{:}$ are interchangeable inside any theorem.

Rule of Existence:
If a term (which can contain variables if they are free) appears in a theorem, then any, several, or all instances of the term may be replaced with some new variable "u" and the $\exists u{:}$ qualifier must be placed in front of the string.

Rules of Equality:
Symmetry: If $r = s$ is a theorem, then so is $s = r$.
Transitivity : If $r = s$ and $s = t$ are theorems, then so is $r = t$.

Rules of Successorship:
Add S: If $r = t$ is a theorem, then so is $Sr = St$.
Drop S: If $Sr = St$ is a theorem, then so is $r = t$.

Rule of Induction:
If u is a variable and $X\{u\}$ is a well formed formula in which u is free, and if both $\forall u{:}<X\{u\}\supset X\{Su/u\}>$ and $X\{0/u\}$ are theorems, then $\forall u{:}X\{u\}$ is also a theorem.

Here, "S0" means "the successor of zero" (or "1") and "SS0" means "2" etc.. T.N.T. uses only natural (non-negative and whole) numbers. The symbols "+" and "·" are simply symbols for addition and multiplication. "$\forall n$: F(n)" means that for all natural numbers "n," the function F(n) must be true; and "$\exists n$: F(n)" means that there exists some natural number "n," such that F(n) is true. (T.N.T. uses the very cumbersome "SSSSSSSSSS0" to express even the relatively small number 10. Here we will be using some very large numbers, so we will make things easier on everyone by using a much less cumbersome numbering system.)

At this point, we appear to have enough capability to generate the entire principles of mathematics (*Principia Mathematica* P.M., by Bertrand Russell and Alfred North Whitehead, published 1910-1913). Historically, as soon as this happened, the question was naturally raised whether or not all mathematical truth could be formalized in this manner.

ASCII "Gödel Numbering":

Then, in 1931, Gödel published the paper which appeared to defeat forever any hope of formally proving all truth. To do this he came up with a way to express statements about T.N.T. in a format where they could be operated upon and tested by T.N.T. itself. His system involved converting statements about numbers into numbers.

Since this was in 1931, any knowledge about the inner workings of a desktop computer was still a good many decades off in the future. Yet Gödel was sufficiently ahead of his time that he invented a way of representing statements as numbers and then a way of performing operations on those numbers. At the time, the system was very difficult for anyone to follow. There may have been only a few people on the planet who were able to perform the mental gymnastics required to understand Gödel's theorem.

But personal computers have changed that for many people. A computer is a completely formal system of operating on simple binary numbers. All of the operations which a computer performs can be thought of as mere extensions to the P.C. and T.N.T.. Yet programs can perform very abstract symbolic operations which appear to have very little to do with the binary numbers which the computer is really manipulating.

To do this, most computers use a sort of "Gödel numbering" system called ASCII (American Standard Code for Information Interchange). The number 97 is the ASCII code for the lower case letter "a." The computer sees an "a" as the binary number "01100001b". (The "b" at the end of the number is to remind us that this is a binary representation for ninety-seven instead of a decimal representation for 1100001.) To save space, this binary number is usually broken into two halves (0110b, 0001b) which are then expressed as two hexadecimal digits (61h). (Letters a-f are used to represent the hexadecimal digits for the numbers 10-15. The "h" reminds us that this number is not a decimal sixty-one but a hexadecimal representation of ninety-seven.)

When you type an "a" into a word processor, the computer represents that symbol with the number "61h." The computer doesn't really care if this represents the number "97," the letter "a," or something entirely different. When it checks your grammar or spelling, it never "knows" it is doing anything more abstract than manipulating simple numbers.

The following is a reasonably familiar representation of what we might find in the memory of a digital computer:

```
00000: c1 61 3a c1 62 3a 28 61 2b 53 62 29 3d 53 28 61      ∀a:∀b: (a+Sb)=S(a
00010: 2b 62 29 2c c1 62 3a 28 53 30 2b 53 62 29 3d 53      +b), ∀b:(S0+Sb)=S
00020: 28 53 30 2b 62 29 2c 28 53 30 2b 53 30 29 3d 53      (S0+b), (S0+S0)=S
00030: 28 53 30 2b 30 29 2c c1 61 3a 28 61 2b 30 29 3d      (S0+0), ∀a:(a+0)=
00040: 61 2c 28 53 30 2b 30 29 3d 53 30 2c 53 28 53 30      a, (S0+0)=S0, S(S0
00050: 2b 30 29 3d 53 53 30 2c 28 53 30 2b 53 30 29 3d      +0)=SS 0,(S0+S0)=
00060: 53 53 30 00 00 00 00 00 00 00 00 00 00 00 00 00 ...  SS0
```

The column on the left identifies the memory address (in hexadecimal) for the first byte of data in each row of memory locations. Each of the sixteen numbers following that address is expressed as two hexadecimal digits which, together, describe the data in that memory location. On the right is a row of symbols which interpret those sixteen data bytes as ASCII characters. In this case, we are using a made-up code "c1" for the symbol "∀". (The ASCII numbering system doesn't really have a code for that symbol.)

If you study the characters in the rows on the right, you will notice that they spell out a series of seven T.N.T. lines which comprise a proof for the T.N.T. theorem "(S0+S0)=SS0." (meaning 1+1=2, see the examples below). This means the computer is representing this proof as a series of numbers. The computer could also chain this series of numbers together into a single large number (which happens to be the normal way that computers store large numbers.) In fact, any string of characters can be represented by a single large number whose value is that of its ASCII characters (in hexadecimal) being strung together into this kind of chain.

Example 1:

(37) (S0+S0)=SS0 ⇒ 2853302b5330293d535330 (hex)

Example 2:

(38) ∀a:∀b:(a+Sb)=S(a+b),	axiom 3
(39) ∀b:(S0+Sb)=S(S0+b),	specification (S0 for a)
(40) (S0+S0)=S(S0+0),	specification (S0 for b)
(41) ∀a:(a+0)=a,	axiom 2
(42) (S0+0)=S0,	specification (S0 for a)
(43) S(S0+0)=SS0,	add S
(44) (S0+S0)=SS0	transitivity (lines 40, 43)

⇒

c1613ac1623a28612b5362293d5328612b62292cc1623a2853302b5362293d53285330
2b62292c2853302b5330293d532853302b30292cc1613a28612b30293d612c2853302b
30293d53302c532853302b30293d5353302c2853302b5330293d535330 (hex)

Of course, this is not the system Gödel used to number his T.N.T. statements (his system involved prime numbers -- see G.P. pp.68+ for a more complete explanation), but it is an obvious way to do it. More importantly, it is a way which may be familiar to many of us; it may be less trouble to keep this kind of "Gödel number" in our heads while we attempt to follow other complicated reasoning at the same time. If you happen to be very familiar with hexadecimal and ASCII, you might be able to "read" the T.N.T. statements directly out of these numbers. Furthermore, it is a simple matter to formally extend T.N.T. to allow substitutions between T.N.T. numbers (like SSSSSSSSS0) and hexadecimal-ASCII Gödel numbers (like 09h). etc. This gives us the ability to to use the T.N.T. rules to operate on this type (hexadecimal) of Gödel numbers.

Proof Pairs: <n → m>

If the number "n" is the Gödel number for a proof of a statement whose Gödel number is "m", then "n" and "m" can be said to constitute a proof pair. Here we will use the symbols <n → m> to designate this relationship. We can also define a T.N.T. extension procedure (more easily pictured as a string-manipulating computer routine) which can determine whether or not "n" describes a proof for "m." Since we know all of the T.N.T. rules, it would be fairly obvious how a skilled programmer might write a computer program to check whether the relationship was valid. The first thing to check might be whether the number "m" exactly matches the end of "n," but the other tests would also be reasonably straightforward. The result of this procedure would be either "true" or "false" and it could be expressed symbolically as <n→m> or ~<n→m> depending on whether or not the result was true.

Example:
<c1613ac1623a28612b5362293d5328612b62292cc1623a2853302b5362293d532853302b6229
2c2853302b5330293d532853302b30292cc1613a28612b30293d612c2853302b30293d53302c
532853302b30293d5353302c2853302b5330293d535330 → 2853302b5330293d535330>

In this example, our "n" is the hexadecimal representation for a proof that "1+1=2", and our "m" is the hexadecimal representation for the statement "1+1=2" itself. In this case, <n→m> is a true statement. At this point we are ready to take an oversimplified look at what Gödel attempted to do.

An Oversimplification of Gödel's String:

Consider the following string of logical terms:

(45) ~∃n: <n → g >

Here we will have "g" represent the number 7ec26e3a3c6ec3**67**3e (hex) which happens to be the Gödel number for the whole string ~∃n:<n→g> (We are using the made-up code "c2" for the symbol "∃"and the made-up code "c3" for the symbol "→") The hexadecimal representation for "g" is emphasized because it will be important for us to keep track of it. This string means: "There is no number 'n' such that 'n' and 'g' constitute a proof pair for the string whose Gödel number 'g' represents 'this string itself'." In layman's terms, this is sort of like saying:

"There is no proof that this statement is a theorem."

Unfortunately, this string contains a circular reference to the symbol "g" (67 hex). In effect, the phrase "this statement" is understood to be replaced by the whole phrase, so we really get something more like:

"There is no proof that (There is no proof that (There is no proof that (...) is a theorem.) is a theorem.) is a theorem."

This isn't a very manageable statement, and it isn't clear that the reason we might not be able to prove this is that we would have to somehow produce "g" itself out of thin air in our proof. What is needed is something which removes the circularity. Here Gödel came up with an amazingly clever solution. He found a way to use "self reference" itself to undo the undesired effects of self reference.

The "Quine" Function: < ↓ >

There was a philosopher named Willard Van Orman Quine who experimented with some interesting self-referencing statements like:

"IS A SENTENCE FRAGMENT," IS A SENTENCE FRAGMENT.

"YIELDS FALSEHOOD WHEN PRECEDED BY ITS QUOTATION,"
YIELDS FALSEHOOD WHEN PRECEDED BY ITS QUOTATION.

Gödel did a similar thing with his numbers. Specifically, he came up with a function which we will call the Quine function (see G.E.B. pp. 435, 445-446 or G.P. pp.90-91). Here we will use <n↓m> to symbolically represent this function. <n↓m> means "m" is constructed by inserting the Gödel number for the number "n" into every location of "n" (interpreted) which symbolizes a free variable. We will not have to worry too much about what the term "free variable" means; "g" happens to be the only "free variable" we will be using here. We can restate this as: construct "m" by inserting the Gödel number for the number "n" into every location of "n" which is marked by the hexadecimal code "67" for "g". (And yes, "n" is going to be a Gödel number itself, so I really mean the Gödel number for a Gödel number here.)

If "n" is 7ec26e3a3c6ec3**67**3e (which is the hexadecimal-ASCII representation for ~∃n:<n→g>), then the Gödel number for "n" is 3765633236653361336336656333**3637**3365 (which is the hexadecimal-ASCII representation for the hexadecimal-ASCII representation for ~∃n:<n→g>). So, by definition of the Quine function "<n↓m>", "m" would have to be:

7ec26e3a3c6ec3[3765633236653361336336656333**3637**3365]3e

Here the square brackets just show where we inserted the Gödel number for "n" into the number "n" itself -- (where the "g" or "67" was).

Gödel's String:

We are now almost ready to construct Gödel's String. We start with the following string (which is still not quite Gödel's String):

(46) ~∃n:∃m:<<n → m>∧<g ↓ m>>

This string has the Gödel number of 7ec26e3ac26d3a3c6ec36d3e263c**67**c46d3e3e (hex). We are using the made-up code "c4h" for the symbol "↓." and 26h (ASCII for "&") for the symbol "∧."

Next we get rid of "g" (the only free variable) by plugging the string's own Gödel number into it:

(47) ~∃n:∃m: <<n→m>∧<7ec26e3ac26d3a3c6ec36d3e263c**67**c46d3e3e↓m>>

This, essentially, is Gödel's string. (We are just using different symbols and numbering.) By definition of the Quine function (↓), we know that "m" is:

7ec26e3ac26d3a3c6ec36d3e263c[37656332366533616332366433 61
33633665633336643365323633633**3637**6334366433653365]c46d3e3e

Which is the hexadecimal-ASCII representation (the Gödel number) for:

(48) ~∃n:∃m: <<n→m>∧<7ec26e3ac26d3a3c6ec36d3e263c**67**c46d3e3e↓m>>

By comparing strings 47 and 48, we see that this is just Gödel's string again. This means Gödel found a non-circular way to refer to "this string." The "↓" (Quine) function achieves this. Even the "67" (meaning "g") in the string has lost its circularity. Now it is just an arbitrary place marker which tells us where the string is to be inserted into itself. (Any symbol would work equally well for this marker.)

Because the Quine function is defined for g

(g being the number 7ec26e3ac26d3a3c6ec36d3e263c**67**c46d3e3e),

we know that <g↓m> must be true for some number "m". Actually, we know that the number "m" is specifically:

7ec26e3ac26d3a3c6ec36d3e263c[3765633236653361633236643361
336336656333366433653236336**3637**6334366433653365]c46d3e3e

(This is the Gödel number for "this string.") Therefore, we know that the Quine function will be "true" when "m" is the the Gödel number for "this string."

This means the right side of the "∧" operator in Gödel's string (47) must be true when "m" is the Gödel number for "this string." Therefore, if the whole string is not a theorem, the failure must be in the other part of it. "<n→m>" must be false when "m" is the Gödel number for "this string". So, this string means "There is no 'n' such that 'n' and 'm' from a proof pair (where 'm' is the Gödel number for this statement)." Or, paraphrased:

(49) "This statement is not a theorem." (of any formal system)

Gödel's string asserts that it is not a theorem of T.N.T. (nor of Principia Mathematica, nor of ***any*** other formal system of mathematics and logic -- see G.E.B. p.447 or G.P. pp 106-108). Gödel managed to make this assertion within the allowed syntax of T.N.T. using only a few obviously valid extensions to it. Now, what does this mean? Gödel's statement asserts that it is underivable (not a theorem), but is it *really* a theorem or not?

Possibility 1: This statement is a provable theorem -- but if so, it would be a self-stultifying theorem. Consequently, it would have to be a false one. This would mean T.N.T. is inconsistent (produces false theorems).

Possibility 2: This statement is not a provable theorem. This is exactly what it asserts, so it would seem that it "must" be a true statement. Therefore T.N.T. would have to be incomplete and could not prove theoremhood for all true statements.

Gödel concluded that this statement had to be true, yet not a theorem -- that, therefore, ***all*** consistent logical systems must be incomplete. Specifically, they must at least fail to prove the "truth" of Gödel's statement (and also, the truth of an infinite number of other statements like it).

The consequences do not end here. We have just decided that Gödel's string (47) (we'll call it G) cannot be a theorem of T.N.T. (even though it is "true"), but neither can its negation ~G be a theorem (since the negation of a true statement must not be a theorem in a consistent logical system). This give us: ~<G∨~G> (see G.E.B. p.449) which is a violation of the rule of excluded middle. Observe the mischief Gödel's theorem has permitted:

(50) ~<G∨~G>
(51) <~G∧~~G> De Morgan's Rule
(52) <~G∧G> Double Tilde Rule

Line 52 gives us what we need (using lines 5-14) to prove that we are the Pope. Only this time, unlike Epimenides, Gödel seems to be quite serious.

Did Gödel Make a Mistake?

Epimenides and Gödel have both made assertions. Epimenides said (line 22):

"This statement is not true."

And Gödel said: (line 49)

"This statement is not a theorem." (of any formal system)

Notice the similarity between the two. These statements become particularly similar when we keep in mind that the goal of a formal system is that *all theorems be true* (and, ultimately, that all non theorems be false). It would appear that, at some level, Gödel might be attempting to state the same paradox as Epimenides. By taking Epimenides' assertion seriously, we can "prove" anything at all to be true. The same appears to be true of Gödel's assertion. However, the correct axiom to extract form the "Emimenides fantasy" is really: (line 36)

< <<p⊃~p> ∧ <~p⊃p>> ⊃ <~p∧p> >

Or perhaps even something like:

<"this statement (within the quotes) is not true" ⊃ absolutely anything>

This is all that is permitted by the fantasy rule.

The "Epimenides" String:

The Epimenides Paradox can be expressed using Gödel's system; but first we will need to define another new function which we will call the "Interpretation" function: «n». Like the "Proof Pair" and "Quine" functions, this is another made-up function. «n» simply means that the Gödel (or ASCII) number "n" is to be interpreted. Thus, if n = 7e70 (hex-ASCII for "~p"), then "«n»" means "~p". This function is as valid an extension to T.N.T. as those other extensions were.

To generate the "Epimenides String" we will start with: ∃m:<~«m»∧<g↓m>>, whose Gödel number, "g" is c26d3a3c7e7b6d7d263c**67**c46d3e3e (hex). (Here we are using the codes "7b" and "7d" for the symbols "«" and "»")

As before, we plug the string's own Gödel number into its "g", and get:

∃m:<~«m»∧<c26d3a3c7e7b6d7d263c**67**c46d3e3e↓m>>

Which, by definition of the Quine function, means "m" must be equal to:

c26d3a3c7e7b6d7d263c[633236643361336337653762
3664376432363363**3637**6334366433653365]c46d3e3e

which is the hexadecimal-ASCII representation (or Gödel number) for:

∃m:<~«m»∧<c26d3a3c7e7b6d7d263c**67**c46d3e3e↓m>>

Which is, as before, just the string whose truth we are questioning. Since we know the Quine function must generate the "m" which is the Gödel number for "this statement", the right half "<c26d3a3c7e7b6d7d263c**67**c46d3e3e↓m>" must be true. This means the veracity of the statement hangs on the "~«m»" part of it (where "m" is the Gödel number for "this statement"). So, its meaning is, effectively:

"This statement is not true." (line 22)

Because it is self stultifying, this statement cannot be a valid theorem of T.N.T.. However, neither is it simply true, or simply false. Instead, being a paradox, it is something which intrinsically violates the rule of excluded middle.

Gödel's Error:

Might Gödel's assertion contain an intrinsic self-contradiction? Because of what follows from it, we ought to be very careful here. We have certainly shown that it cannot be a provable theorem (and therefore be false by its own assertion), but how carefully did we test the idea that it was a "true" non-theorem? We never tested the possibility that it might be a *paradoxical* non-theorem.

Since Gödel's statement is certainly a non-theorem, and since it asserts it is a non-theorem, then it might seem obvious that it must be a "true" statement. Unfortunately, real truth is not always this simple. We must remember that we have used some kind of *reasoning* (asserted to be valid) for sorting this statement into the "truth" category. Since Gödel's theorem applies to ***all*** formal logical systems, It might be of interest to consider whether or not our reasoning (when we decided Gödel's statement was true) might itself be considered a *formal* logical system. If so, this would make Gödel's statement a "theorem" of whatever reasoning system we happened to be using. Since it is now a theorem, this would make it a false statement -- which then turns it back into a non theorem and a true statement. In short, it would not simply be true or false; it would be paradoxical -- like Epimenides' statement. For this reason, we must presume that Gödel's claim cannot be supported *formally*.

In order for Gödel's claim to hold up, it's supporting arguments must fall short of formality in some way. Logical arguments which cannot be made *formally* may not really be all that logical. Let's take a careful look at the "logic" we were using when we identified Gödel's statement as being "true." In essence, we assumed: "Any statement which asserts the truth about what "has been" (in the past) concluded about it, "is" (in the present and future) still a "true" statement." We failed to account for the circularity or "oscillating" nature of paradoxical statements. This is not valid reasoning. So, whether or not our reasoning might have been considered "formal," it was certainly faulty.

Like Epimenides' statement, Gödel's statement intrinsically violates the law of excluded middle. We ought to have treated Gödel's entire derivation as a formal fantasy. His string was merely assumed for the sake of argument; no reason was presented why we should take it any more seriously than Epimenides' paradox. A more formal conclusion would be what the fantasy rule permits; something like:

<"this statement (within the quotes) is not a theorem" ⊃ all consistent logical systems must be incomplete>

Conclusion:

From the preceding argument, it appears to me that Gödel's string is intrinsically paradoxical. This puts it in the same category as Epimenides' string. If we fail to treat it properly, as the premise for a formal fantasy, then we can conclude anything we choose -- including that, "All consistent logical systems are incomplete." (It might still be true that all consistent logical systems are incomplete, but if so, it would seem that it has not yet been proven.)

Please send questions and comments to: stonerdon@yahoo.com