Godel's incompleteness theorems

Andrew Zhao

March 2017

1 What is Godel's theorem?

Godel's incompleteness theorems describe the limitations of formal systems. Published in 1931, Godel's paper contains two main theorems about the properties of formal systems. Informally, the two theorems state that:

1. Every formal system strong enough to represent arithmetic cannot be both consistent and complete.
2. A formal system representing arithmetic cannot prove it's own consistency.

To discuss these theorems, let's first take a look on what a formal system is.

2 Formal Systems

Formal systems are comprised of theorems, which are generated by axioms and rules of inference.

Definition 2.1. A theorem is a statement that is true in a particular formal system: in other words the theorem can be derived from the system's axioms and rules of inference.

Definition 2.2. An axiom is a statement in a formal system that is true by default in the formal system.

Definition 2.3. A rule of inference of formal systems allows us to generate new theorems in a formal system.

For example, consider what I will call the pq formal system. Each theorem in this system is composed of p's, q's and hyphens. For our axioms, we let x p - q x- be a theorem, where x is an arbitrary number of hyphens. Our rule of inference is that if x p y q z is a theorem, then x p y - q - z is a theorem, where x, y and z are strings of hyphens.

3 Consistency and Completeness

Consistency and completeness are two definitions that play an important role in Godel's theorem.

Definition 3.1. A consistent system is a formal system where it is impossible to derive a statement and it's negation simultaneously.

Definition 3.2. A complete system is a formal system where given any statement satisfying syntax requirements of the formal system, one can prove either that the statement is a theorem or that the statement is not a theorem.

Turns out that the pq formal system that I gave above is both consistent and complete(one can verify that easily). So it might seem that this system disproves Godel's theorem. In fact, Godel's theorem also requires the system be strong enough to represent arithmetic over the integers. This requirement, as will be seen later, is fundamental because the construction of the Godel string used to prove this theorem requires certain functions in arithmetic.

4 Idea of the proof

Consider the following sentence: This sentence is a false statement.
Lets assume that this sentence is a true statement. Then the statement of the sentence implies that the sentence is false, a contradiction. Let's now assume that the sentence is false. Then the statement of the sentence is true, so the sentence is true. Therefore, the sentence cannot be proven true nor false.
Now we know that this self-referential sentence cannot be true nor false, all we have to do is import this sentence into the formal system. Then we have constructed a statement that cannot be proved, proving Godel's theorem for that formal system.

5 Constructing the statement

However, importing the statement into the formal system has certain challenges. For example, how is one going to write a statement that talks about itself using symbols that can only talk about numbers? Indeed, the string does talk about properties of natural numbers, but in a way such that it ends up talking about itself. Constructing the string used to prove Godel's theorem is where the beauty and genius of his proof lies. Let's construct a self-referential statement using Godel Escher Bach's TNT formal system to show how one would construct the string in any formal system satisfying the conditions of the theorem.
First, let's find a way to get statements in a formal system to talk about other statements. To do so, we note that since TNT only talks about numbers, we can express each symbol in TNT using a number. For example, we can denote = as 111, free variable a as 262, Successor function S as 123, 0 as 666, ∃ as 333, so on. This type of numbering is known as Godel numbering, and is the first step of our proof by allowing us to talk about statements in TNT. To do so, we introduce the definition of a TNT-number, which can be defined as a Godel number that can be derived in the system. For example, let's say using TNT-notation that S0=0. Using the numbering above, we can claim that 123 666 111 666 is a TNT-number. But here, we have a string that is talking about a string in TNT.
Let's formalize the idea of TNT-strings talking about numbers that represent TNT-strings by using proof pairs.

Definition 5.1. Two natural numbers a and b form a TNT-proof pair if and only if a is the Godel number of a derivation whose bottom line is the string with Godel number b.

Now we claim that the proof pair property is representable as a formula in TNT, so we can state that two numbers are proof pairs in TNT. To do so, we note that we just have to turn them back into symbols and check whether m is consistent and whether n is also consistent. Since it is possible to check it in a finite amount of time, there must exist some formula in TNT-symbols to determine proof-pair property. This allows us to write ∃a: TNT-PROOF-PAIRa,a', which states that there exists a number a such that if a' is a Godel number, a and a' form a TNT-proof pair.
But all we have done is formalize the notion of referencing other strings in TNT, but we have not come up with a way to get this reference back to the string. To do so, we come up with a funky function. This function takes in a string of TNT, looks for all of the free variables in the string, and substitutes the string's Godel number into the string. For example, consider the statement a = 0, with Godel number of 262 111 666. When substituting, we use the successor function, or 262111666 S's. So the function, which we call MAGIC, or MAGIC(c) for a Godel number c, will give a Godel number of 123123123123.... 111 666, where there are 262111666 S's.
In fact, let's formalize the idea of MAGIC. We say that MAGICa", a' returns true if a' is the Godel number of the result of applying the MAGIC function to the formula with Godel number a". and false otherwise. Here, note that we are still talking about numbers, so it is legal in TNT to have this statement.
Now let's see why I called MAGIC magic. To achieve a self-referential statement, we are going to use the MAGIC function on a string that already talks about MAGIC. So let's combine MAGIC with TNT-PROOF-PAIR to create a statement x which states: (or does not)∃ a, ∃ a such that TNT-PROOF-PAIRa,a' and MAGICa",a'. But this statement is true since we can find any a" to satisfy the second statement. To make this string an actual statement, we apply that MAGIC function to this string to get ∃ a, ∃ a such that TNT-PROOF-PAIRa,a' and MAGICSSSSSSSSS...0,a', where there are x S's by definition. This is the Godel string, which we denote by G.

First, let's simplify the meaning of this string. In English, it states that "There do not exist numbers a and a' such that they are proof pairs and a' is the number that is given by applying the MAGIC function to u. However, by definition of MAGIC, there must exist a number that satisfies the second condition, so there is a problem with the first statement, implying that the string states there is no number a that forms a TNT-PROOF-PAIR with MAGIC(x). By definition of proof pair, the string states that the formula with Godel number MAGIC(x) is not true.
Now for the final blow. By the way we constructed G, G's Godel number is MAGIC(x). So the string is stating that itself is not true!

6 What about the Second Theorem?

The second Godel theorem states that a system cannot prove it's consistency using a weaker system than itself. In other words, if one wants to prove a formal system's consistency, one cannot reduce that to proving a weaker system's consistency. Proving this theorem is similar to proving the first theorem: we construct a statement in this system which represents consistency, and then show that this statement cannot be proved.

7 Notes on Theorem

When published, Godel's two theorems shocked the mathematical community. At that time, people believed that all mathematics could be derived error-free from logic. This belief was so widespread that Bertrand Russell and Alfred North Whitehead wrote a book titled Principia Mathematica which supposedly derived all mathematics using only logic (and took 350 pages to prove that $1 + 1 = 2$). Godel's theorems proved that such an attempt was impossible.
Practically however, Godel's theorem seemed to have little impact. Indeed, the Godel string was so contrived to suit it's purpose that there was no purely mathematical analogue for the Godel string. However, turns out that the Continuum Hypothesis, or the statement that no set has cardinality between the cardinality of the set of integers and the set of reals, cannot be proved true nor false using axioms of set theory. This type of theorem, foreshadowed by Godel's theorems, had much more impact on actual mathematics than Godel's theorem.
There are several theorems in computing and math that use similar strategies to prove the impossibility of some property. In particular, the halting problem, or the problem of determining whether a arbitrary program will halt on a particular input, has been shown to be impossible to solve by using the paradox in self-referentially.