Mathematical Induction
- a miscellany of
theory, history and technique

Theory and applications for advanced secondary students

Part 4

Peter Haggstrom

The building blocks of GÖdel's Theorem

The foundations of Gödel's Theorem (ie his undecidability theorem) are to be found in about 50 papers contained in a book by **Jean van Heijenoort:, "From Frege to Gödel: A Source Book in Mathematical Logic, 1879 - 1931".** Every serious logic student has read this book. Van Heijenoort is worth mentioning even if only for his colourful past which is described by **Benjamin H Yandell in his book "The Honors Class: Hilbert's Problems and Their Solvers", A K Peters, 2002, page 66.** I quote in full:

" I assumed van Heijenoort was merely a logician as I gratefully pored over his book and was astounded when I learned that as a young man, in the 1930s, he had followed Leon Trotsky from Turkey to France to Norway to Mexico, as his body guard and secretary. Anita Feferman's "*From Trotsky to* Gödel*": The Life of Jean van Heijenoort"* tells van Heijenoort's remarkable story. Trotsky and his camp were pursued by Stalinist agents; van Heijenoort always packed a gun. Van Heijenoort had an affair with an artist Frida Kahlo. He epitomized cool and was attractive to women, and this continued even after he became a logician, He had grown tired of the rigors of life with Trotsky and traveled to New York in 1939 on a somewhat vague mission. (In a later period he hung out with painters Jackson Pollock, Willem de Kooning and Franz Kline.). He blamed himself for Trotsky's assassination, because he felt he would have recognised the assassin, a Spaniard named Ramon Mercader posing as French - speaking Belgian, was not a native speaker of French. (Van Heijenoort was French, not Dutch). Van Heijenoort himself was murdered by his fourth or fifth wife, "depending in how one counts", in

Mexico City in 1986 - three bullets in the head as he slept".

Ironically, Trotsky was killed in Mexico too - the method being an ice pick through the skull rather than three bullets to the head.

Before we go any further let us develop a "toy" formal system as follows. In essence the system is simply an alphabet coupled with some rules about how you create strings of symbols (ie "formulas") and make deductions from those strings of symbols. You need some axioms as basic building blocks to get you going and some rules of inference which are essentially the logic "engine". Note that we are not attaching any meaning to the symbols at this stage - rather all that we are doing with such systems is seeing how to generate valid strings of symbols in the system. Such a system should be able to tell us whether some string is an immediate consequence (ie follows directly by an application of the relevant rules) of another string.

Let Z be the system defined this way:

Alphabet: Δ ■
Formulas: Any finite string of symbols from the alphabet of Z that begins with a 'Δ' is a formula of Z. Nothing else is a formula of Z.
Axiom: Δ■■■
Rule of inference: Any formula of Z whose last two symbols are a 'Δ' and a '■', in that order, is an immediate consequences in Z of any formula of Z whose first two symbols are a 'Δ' and a '■', in that order. So, for instance, 'Δ■■■ΔΔ■' is an immediate consequence in this toy formal logic system of 'Δ■ΔΔΔ■Δ'. Nothing else is an immediate consequence in Z of anything. Note that the rule of inference can be applied to the axiom.

Is 'ΔΔ■' an immediate consequence of 'Δ■■■'? It is an immediate consequence because 'Δ■■■' is the axiom and the rule of inference says that anything with a trailing 'Δ■' is an immediate consequence.

Note that 'Δ■■Δ' is not an immediate consequence of 'Δ■■Δ' because only formulas which end with 'Δ■' can be an immediate consequence. Some strings of symbols are not valid formulas eg '■■ΔΔ■' because it starts with a '■'.
There are also strings which do not admit an immediate consequence eg 'ΔΔΔΔΔ'. This string is a valid formula because it starts with a 'Δ' but it cannot have an immediate consequence because our rule of inference is that only formulas which begin with 'Δ■' can have immediate consequences.

This toy formal system is pretty boring but it does capture some of the fundamental concepts of how number systems were developed as formal systems of logic. Basically you have symbols such as "∀", "~", "(", ")" , "x", "y", "⇒", etc which are stringed together to form well formed formulas (ie ones that make sense based on certain rules of composition of symbols) and then you have a series of well formed formulas connected in ways which give rise to proofs. There are axioms (which are accepted as true) and there are rules of inference (ie how to get from A to B in a logical sense). An example of a rule of inference is *modus ponens* ie " (p & (p⇒q)): ⇒: q". What this says is that if raining implies wetness and you know it is raining, then it must be wet.

As noted above, induction is fundamental to systems of logic. Given that mathematical logic systems seek to develop the structure of number systems this is not surprising. Recall that the principle of mathematical induction can be stated as the following general proposition:

Suppose that (i) 0 has the numerical property P and suppose (ii) that for any natural number n, if it has property P, then its successor n+1 also has property P.
(iii) Then we can conclude that every number has property P. Since the successors of 0 are the only natural numbers we pick up all natural numbers - there are no "stray" numbers that can cunningly insinuate themselves into the chain.

Giuseppe Peano, an Italian mathematician and logician published a list of axioms in 1889, which form the basis for a simple system of arithmetic. These axioms had already been formulated by the German mathematician Richard Dedekind in 1888 but everyone in the logic world refers to the axioms as Peano axioms rather than Dedekind axioms. Life is unfair. The axioms were as follows:

Axiom 1: $(\forall x)(0 \neq Sx)$ Here S is the successor function. What this axiom says is that 0 is not the successor of any number.

Axiom 2: $(\forall x)(\forall y)(Sx = Sy \Rightarrow x = y)$ This says that if the successor of x equals the successor of y then x must equal y.

Axiom 3: $(\forall x)(x + 0 = x)$ This just says that adding 0 to anything doesn't change the sum.

Axiom 4: $(\forall x)(\forall y)(x + Sy = S(x + y))$ This says that $x + (y + 1) = (x + y) + 1$

Axiom 5: $(\forall y)(y \times 0 = 0)$ This says that anything multiplied by 0 is 0

Axiom 6: $(\forall x)(\forall y)(x \times Sy = (x \times y) + x)$ This says that $x (y +1) = x y + x$

Induction schema: $(\{ \varphi(0) \& (\forall x) (\varphi(x) \Rightarrow \varphi (Sx)) \} \Rightarrow (\forall x)\varphi(x)$ This says that if $\varphi(0)$ holds (ie has some value) and whenever $\varphi(x)$ holds, $\varphi(x+1)$ holds, it follows that φ holds for all x.

The Peano system and equivalent systems are very basic because they only cover the successor function, addition and multiplication. Such a system is quite primitive and much thought in the mathematical logic community was given to developing a richer functional structure. There is a wide class of functions called "recursive functions" which form the basis for richer systems of logic. A recursive function is essentially one that is defined in terms of itself. The factorial function is an example; n! = n (n-1)(n-2)...2.1 The recursive nature can be seen when n! is defined as follows :

$n! = f(n) = n\, f(n-1)$ for $n \geq 1$ and $0! = 1$

Thus $3! = 3\, f(2) = 3.2\, f(1) = 3.2.1\, f(0) = 3.2.1$

Another example is the Fibonacci function: $f(n) = f(n-1) + f(n-2)$ for $n > 1$ where $f(0) = 0$ and $f(1) = 1$

Using the successor function S we can define the factorial function as follows:

(1) $0! = S0 = 1$

(2) (Sy)! = y! × Sy. This says that to get the factorial of the successor of y you multiply y! (which has to be already calculated) by the successor of y. Thus (S2)! = 2! × S2 = 3 × 2! = 3 × (S1)! = 3 × 1! × 2 = 3.2.1

To get a more complicated function consider x^y. This can be defined as follows:

(3) $x^0 = 1$

(4) $x^{Sy} = x^y \times x$. Thus $x^{S3} = x^3 \times x = x^4$

The point of all of this is that using primitive recursive functions as building blocks it is possible to combine them in general ways in a chain of definitions to get something more complicated. The beauty of recursion is that if you have something complicated, you can systematically (albeit tediously) go backwards through the definitional chain to get to the very basic starter functions. Thus in the case of the factorial we can view it this way:

f(0) = 1
f(Sy) = h(y,f(y)) where h is some function we already know about prior to defining f. We can define f(Sy) as (Sy)! = y! × Sy as before. Using the new function h we have f(Sy) = h(y,f(y)) where h(y,u) = u × Sy. Let's build the chain from f(0) and see how it all works since the abstract way of defining h obscures things a bit:
f(0) = 1
f(1) = f(S0) =h(0,f(0)) = h(0,1) = 1 × S0 = 1 × 1 = 1
f(2) = f(S1) = h(1,f(1)) = h(1,1) =1 x S1 = 1 × 2 = 2
f(3) = f(S2) = h(2,f(2)) = h(2,2) = 2 × S2 = 2 x 3

etc

Thus f(3) = 3! = 3.2.1 and you can see ow you could work backwards knowing the form of h.

Two place functions such as addition, multiplication and exponentiation can be defined via a general scheme of the following form:

f(x,0) = g(x)
f(x,Sy) = h(x,y,f(x,y)) where g and h are functions we already know.

To cut a long and very complicated story short, by generalising functions in the manner described above out of primitively recursive building blocks the idea is that you are able to build a wide variety of key functions in the context of a rich number system. Gödel's Theorem involves proofs of such matters but I will not dwell on the details. The other attraction of primitive recursive functions is that they are computable and that suggests that you might be able to devise a mechanical system for working out all the theorems (true statements) of a formal system built out of such building blocks. While all primitive recursive functions are effectively computable (ie susceptible to a mechanical process of derivation) the converse is not true. **The nub of primitive recursion is the concept of a for-next loop in computing. If you can construct s series of for-next loops which only invoke already known primitive recursive functions, then you have demonstrated that the function is effectively computable.**

Thus the broad drift of Gödel's theorem is that:

1. you work out a how to systematically describe a sufficiently rich system of number theory to be interesting (ie give rise to significant propositions like Fermat's Last Theorem say) and recursive functions play an absolutely fundamental role in that context.

2. you arithematize the system - in other words you translate the symbols and strings of symbols in a unique way and you thereby reduce statements about strings of symbols (and strings of strings of symbols making up more complex deductions) to statements about numbers. **This is where Gödel's numbering system comes in and it relies upon the Fundamental Theorem of Arithmetic for its uniqueness properties**.

3. **if this system is sound (ie it does not produce falsehoods because the axioms are true and the rules of inference are truth preserving), then there is a Gödel sentence G which is true if and only if it is not provable. Gödel demonstrated that sentence.** If G could be proved within the system the theory would prove a false theorem which contradicts the assumption of soundness. So G is not provable within the system. Hence G is true and so ¬G is false. Hence ¬G cannot be proved in the system either. So G is a formally undecidable sentence of the system.

Gödel's theorem is not easy to follow without quite a bit of training in formal logic systems. Raymond Smullyan's book **"The lady or the tiger? and other logic puzzles", Oxford University Press, 1982** gives a systematic treatment in an accessible way of the basic ideas. One of the better books which gives a detailed and systematoc approach to the explanatioon of the theorem is **Peter Smith, "An Introduction to Gödel's theorem", Cambridge Univeristy Press, 2007**.

Gödel's approach to arithmetization of number theory

This is the crux of Gödel's theorem since he correlated syntactic properties of strings of formulas that comprise a proof with purely numerical properties. Thus the buildings blocks of his logical system are effectively translated into primitive recursive numerical properties. For instance Atom(n), Wff(n) and Sent(n) respectively hold when n codes for an atomic wff, a wff or a closed wff ie a sentence. A wff is a well formed formula ie something that complies with some strict formation rules. An atomic wff is usually just symbol like the letter 'p'.

But that is all a lead up to the numerical relation Prf(m.n) which holds when m is the code number for a derivation (or proof) in the system of the sentence with code number n. We shall soon see how Gödel did this arithmetization.

Now I am cutting lots of corners here but the next big concept that Gödel employed was to take a wff which looks like this: φ(y) where y is a free variable or place holder. Gödel substitutes the numeral for φ(y)'s own code number in place of the free variable. This process is a form of a "diagonalisation" argument that was made famous by Cantor in his proof that the reals were uncountable. In effect a wff is formed which indirectly refers to itself via the Gödel coding. This technique forms the foundation for generating a sentence which says in effect " I am unprovable in the system".

Now all this sounds very circular and smacks of the so-called Liar Paradox. The Liar Paradox has various forms such as "This sentence is false" so that, if true it is false and if false, it is true. The structural problem is of self-reference in the context of asserting a property of being a property that does not apply to itself. Does this apply to itself? Bertrand Russell and Alfred North Whitehead worked out how to deal with this in *Principia Mathematica*. In essence they developed a stratified theory of types (based on work done earlier by the German logician Frege - pronounced "Fray-ga"). You need to

be able to distinguish between classes and classes-of-classes and classes-of-classes-of-classes and so on. You then insist that classes at level *l* can only have as members inhabitants of level *l-1*. Talk about classes amounts to talk about their defining properties. Thus being wise is a level 1 property and if Socrates is a level 0 item it makes sense to attribute the level 1 property to Socrates ie to assert that Socrates is wise. You could go further and think of the property of having some instances which would be a level 2 property since it would make sense to attribute that property to wiseness ie the property of being wise has some instances. However, if you sought to attribute the property of having some instances to level 0 (ie Socrates) you would be saying that Socrates has some instances and this doesn't really make any sense. Similarly, "This sentence is false" refers to a sentence but not one of a lower type in the hierarchy and that is what causes the consistency problems. The self-reference problem does not occur with this sentence for instance: The statement "1 +1 = 3" is false.

The basic reason that Gödel's approach does not involve problems of self-reference is that he developed the full machinery of primitive recursive functions in his paper. Gödel spent a lot of time developing the relevant preliminary steps. Looked at from a high level if all the building blocks are primitive recursive in character then conceptually (and practically in a monumentally tedious way) you can go backwards down through the chain of deductions. Self-reference is effectively avoided by such an approach.

How does one actually go about the arithmetization process? Gödel said the following: "Furthermore, variables of type n are given numbers of the form p^n (where p is a prime number > 13). Hence, to every finite series of basic signs (and so also to every formula) there corresponds, one- to-one, a finite series of natural numbers. These finite series of natural numbers we now map (again one-to-one correspondence) on to natural numbers, letting the number $2^{n_1} . 3^{n_2} ....p_k^{n_k}$ correspond to the series $n_1, n_2,,,,n_k$, where p_k denotes the k-th prime number in order of magnitude. A natural number is thereby assigned in one-to-one correspondence, not only to every basic sign, but also to every finite series of such signs." **Kurt Gödel, "On formally undecided propositions of Principia Mathematica and related systems", Dover Publications (1962), page 45.**

You need to define the symbols and how they are mapped to numbers. Because there are certain logical equivalences you get into issues of minimal numbers of logical operators. For instance, to say that "Not all numbers are even" could be expressed as "There is at least one odd number" and symbolically you could do this as "(∃x)(x is an odd number)" or "~(∀ x)(x is an even number)" (ie it is not the case that every number is even). Thus you don't need both symbols "∀" and "∃". Gödel mapped his basic symbols this way in his paper:

"0" → 1,
"f" → 3,
"~" → 5,
"v " → 7,
" ∀ " → 9,
"(" → 11,
")" → 13

Rather than follow Gödel let's map our basic symbols to numbers as follows:

$\neg$	1
$\bigwedge$	3
$\bigvee$	5
$\rightarrow$	7
$\leftrightarrow$	9
$\forall$	11
$\exists$	13
==	15
(	17
)	19
0	21
S	23
+	25
$\times$	27
x	2
y	4
z	6

etc etc

Let the expression e which comprises a sequence of k + 1 symbols and/or variables $s_0, s_1,,,, s_k$. The Gödel number (GN) for this sequence is determined by taking the code number $c_i$for each $s_i$in turn, using c_i as an exponent for the $i + 1^{th}$ prime number $\pi_i$and then multiplying to get $\pi_0{}^{c_0}.\pi_1{}^{c_1}.\pi_2{}^{c_2}.....\pi_k{}^{c_k}$.

The symbol 'S' (for the successor function) has the GN 2^{23} (the first prime raised to the power of the code obtained from the table above). Thus SS0 (ie the number 2) has GN $2^{23}.3^{23}.5^{21}$ (the product of the first 3 primes raised to the appropriate powers from the table). The GN for the well-formed formula $\exists y(S0 + y) = SS0$ is the monstrously large number:

$2^{13}.3^4.5^{17}.7^{23}.11^{21}.13^{25}.17^4.19^{19}.23^{15}.29^{23}.31^{23}.37^{21}$

To decode this number is of course not quick but it is capable of a mechanical process and that is the point. The crucial part of Gödel 's proof relates to the concept of Prf(m.n) which holds when m is the code number for a derivation in the system of the sentence with code number n ie a proof. A proof is a sequence of wffs or other expressions:

$e_0, e_1, ..., e_n$ where we code each $e_i$by a GN $g_i$which yields a sequence of numbers:
$g_0, g_1, ..., g_n$

You can view a practical demonstration of how Gödel numbering works by going to this link:
http://demonstrations.wolfram.com/Goedelization/

So where does the Fundamental Theorem of Arithmetic fit in and what is it anyway? As already noted above, the Fundamental Theorem of Arithmetic underpinned the uniqueness of Gödel's numbering system. Because the numbering was unique you could be sure that the mapping of numbers to symbols, strings of symbols etc was unique ie there is a 1 to 1 correspondence between the two.

The Fundamental Theorem of Arithmetic: For each integer n > 1, there exist primes $p_1 \leq p_2 \leq p_3 \leq ... \leq p_r$ such that n = $p_1\ p_2 \ldots \ldots p_r$ and this factorisation is unique.

The uniqueness of the factorisation was used by Gödel in his numbering system to achieve a 1-1 correspondence between strings of symbols and their Gödel numbers.

The proof of this theorem turns upon demonstrating that each integer has at least one prime factorisation. This involves induction. You then need to prove uniqueness and that too involves induction.

The prime factorisation of 12, for instance, is 2.2.3 or 2.3.2 or 3.2.2. In other words the unique factorisation is $3.\ 2^2$. Incidentally 1 is not treated as a prime as it would pointlessly complicate the proof of this theorem and pretty much everything else.

There are some preliminary number theoretic results that are needed.

(1) In order that there exist integers x and y satisfying ax + by = c, it is necessary and sufficient that d | c where d = gcd(a,b) = the greatest common divisor of and and b

Proof:

Sufficiency (ie we assume that d | c where d = gcd(a.b) and seek to prove that there are integers x and y satisfying ax + by = c

By assumption d | c so c = kd for some integer k. Now Euclid's Division Algorithm says that for any integers k (> 0) and j, there exist unique integers q and r such that 0 ≤ r < k and j = qk + r. This forms the basis for the following propositon: If d = gcd(a,b) then there exist integers x and y such that ax + by = d. In effect one uses an inductive argument in the context of the Division Algorithm to establish the result. I won't bother proving that but it justifies the conclusion that if c = kd where d = gcd(a,b) there exist u and v such that :

au +bv = d

Hence a(uk) + b(vk) = dk = c

In other words x = uk and y = vk are the required solutions to ax + by = c

Necessity: (ie we assume that ax + by = c and seek to establish that d | c where d = gcd(a,b).

Suppose a = ed and b = fd then c = edx + fdy = d (ex + fy)

Thus d | c.

(2) If a, b and c are integers where a and c are relatively prime, and if c | ab, then c | b.

Proof

Since a and c are relatively prime gcd(a,c) = 1 and (1) above implies the existence of integers a and y such that cx + ay = 1. Hence cbx + aby = b. Now since c | ab there is a k such that ab = ck and when this is substituted we get:
cbx + kcy = b

ie c(bx + ky) = b and this means that c | b.

(3) If a and b are integers, and p is a prime such that p | ab, and p $\nmid$ a then p | b

Proof

If p $\nmid$ a then a and p are relatively prime so that gcd (a,p) = 1 (the only positive divisors of p are 1 and p). Putting c = p in (2) we get p | b.

(4) If p is a prime such that p | $a_1 a_2 \ldots a_n$ then there exists some i such that p | a_i

Proof

This involves an inductive proof. For n =1 the claim is clearly true. Note that when n =2 (4) is essentially a restatement of (3). Assume that the assertion holds for all n ≤ k. Then for n = k +1 consider the following:

p | $(a_1 a_2 \ldots .. a_k)\, a_{k+1}$

Now, using the induction hypothesis, by (3) either p | a_{k+1} in which case i = k+1 or p | a_i for some i where 1 ≤ i ≤ k. This establishes the result by the principle of induction.

The proof of the Fundamental Theorem of Arithmetic

First we need to establish that each integer has at least one prime factorization. To do this we assume that each integer m where 1 < m ≤ k can be factored into primes. This is our induction hypothesis.

Now take k +1. It is either prime or it isn't. If it is prime then its prime factorization is just itself. If it is not prime then it is composite ie k +1 = ab where 1 < a < k+1 and 1 < b < k +1. Since 1 < a ≤ k and 1 < b ≤ k, both a and b have prime factorizations by the induction hypothesis. These factorizations are, say,

a = $p_1 p_2 \ldots . p_s$ and b = $p'_1 p'_2 \ldots p'_t$

Hence k +1 = $p_1\, p_2\, ...\, p_s\, p_1' p_2' ...\, p_t'$

This means that k+1 has a prime factorisation and by the principle of induction every integer greater than 1 has a prime factorisation

Now for uniqueness which again involves an appeal to induction. We assume that for each integer m such that $1 < m \leq k$ has a unique prime factorisation. Now suppose that k +1 has two prime factorisations

k+1 = $p_1\, p_2\, ...\, p_u = p_1' p_2' ...\, p_v'$

where $p_1 \leq p_2 \leq ... \leq p_u$ and $p_1' \leq p_2' \leq ... \leq p_v'$

Since p_1' divides k+1 it must divide $p_1\, p_2\, ...\, p_u$ and so by result (4) above, p_1' must divide p_i for some i. But both p_i and p_1' and are primes so that $p_i = p_1'$

Now reverse this argument and focus on p_1 and p_j' for some j to get $p_1 = p_j'$. You then have that $p_1 = p_j' \geq p_1'$ (by the way we constructed the terms) and, similarly, $p_1' = p_i \geq p_1$. Therefore we have that $p_1 \geq p_1' \geq p_1$ which means that $p_1 = p_1'$. Thus $\frac{k+1}{p_1}$ is an integer not exceeding k and $p_2\, p_3\, ...\, p_u = \frac{k+1}{p_1} = p_2' p_3' ...\, p_v'$ Hence by the induction hypothesis, u = v, $p_2 = p_2'$, ..., and $p_u = p_u'$