Gödel's Legacy: What is a Proof?

April 24, 2007

Hilbert's Entscheidungsproblem

"The Entscheidungsprobelm is solved when one knows a procedure by which one can decide in a finite number of operations whether a given logical expression is generally valid or is satisfiable. The solution of the Entscheidungsproblem is of fundamental importance for the theory of all fields, the theorems of which are at all capable of logical development from finitely many axioms."

D. Hilbert, W. Ackermann
Grundzüge der theoretischen Logik, 1928

Hilbert was looking for an algorithm that could, given any statement in mathematics, determine whether that statement was true.

Such an algorithm would be the holy grail of mathematics – one could simply feed any statement to the algorithm to determine if it is true. All of mathematics would be reduced to finding interesting questions to ask the machine, and improving the efficiency of the machine.

Before we look into this problem, what exactly is meant by "logical development from finitely many axioms"?

Ordinary english is unacceptable for parsing by machines. Hence, the first step is to define a formal language in which to express mathematical ideas.
To avoid confusion, one must be careful to distinguish the object-language under study from the meta-language in which we reason about the system.
If you take Chinese at CMU, Chinese is the object language and English is the meta-language. For this lecture, we will study several different formal object-languages for representing mathematical ideas. Our meta-language is English, used together with convenient mathematical notation.

We are going to have variables in both our meta-language and object-language. In fact, we might have a variable in the meta-language whose value is a variable of the object-language. For example, suppose we want to denote an arbitrary variable of the object-language. We could call it "x". But what if the object-language contains x as one of its symbols? It's not clear if by "x" we mean the symbol x of the object language, or if x is just a variable of our meta-language.

Variables of the meta-language whose values are symbols or words of the meta-language are called syntactical variables, and to avoid confusion, we'll use boldface for all such variables. So x always refers to a specific variable of the object language, while $\mathbf{x}$ is a variable of the meta-language, whose value may be a variable of the object language.
We might have $\mathbf{x} = x$ or $\mathbf{x} = y$ or $\mathbf{x} = z$.

Once one has a language down, the next step is to be precise about what it means to reason in that language. We must be careful not to forget ultimate goal of this system being computable: the reasoning should be systematic enough to be carried out by machines.

- Certain words of the object-language will be designated axioms. These will serve as our basic truths.
- The system will have rules of inference, to derive new truths from already established ones.

All of this is most easily explained with an example. We will define a system $\mathcal{P}$, called propositional calculus. This system will serve as an example of how one can axiomatize logical reasoning.

The alphabet of $\mathcal{P}$ consists of:

- The symbols $[,], \sim, \vee$
- Propositional variables: $p, q, r, s, p_1, q_1, r_1, s_1, \ldots$

A formula is a finite string of symbols. Of these, we shall be exclusively interested in well-formed formulas (wffs), defined inductively:

- If **p** is a propositional variable, then **p** is a wff.
- If **A** is a wff, then $\sim$ **A** is a wff.
- If **A**, **B** are wffs, then [**A** $\vee$ **B**] is a wff.

Which of the following are wffs?

- p
- $\sim\sim\sim p$
- $]]p\vee\sim$
- $p \vee q$
- $[p \vee q]$
- $\sim [\sim p \vee p]$
- $\sim \mathbf{A}$

It's helpful to introduce some abbreviations in the meta-language.

- $[\mathbf{A} \supset \mathbf{B}]$ stands for $[\sim \mathbf{A} \vee \mathbf{B}]$
- $[\mathbf{A} \wedge \mathbf{B}]$ stands for $\sim [\sim \mathbf{A} \vee \sim \mathbf{B}]$
- $[\mathbf{A} \equiv \mathbf{B}]$ stands for $[[\mathbf{A} \supset \mathbf{B}] \wedge [\mathbf{B} \supset \mathbf{A}]]$.

Note we could have instead added $\supset, \wedge, \equiv$ to the alphabet of $\mathcal{P}$. But it will be useful for the language of $\mathcal{P}$ to be as simple as possible. We use $\supset, \equiv$ instead of $\Longrightarrow$, $\Longleftrightarrow$ so the latter can be used in the meta-language without confusion.
We will also omit the outermost brackets when writing wffs.

The axioms of $\mathcal{P}$ consist of all wffs of the following forms:

1. $[\mathbf{A} \vee \mathbf{A}] \supset \mathbf{A}$
2. $\mathbf{A} \supset [\mathbf{B} \vee \mathbf{A}]$
3. $[\mathbf{A} \supset \mathbf{B}] \supset [[\mathbf{C} \vee \mathbf{A}] \supset [\mathbf{B} \vee \mathbf{C}]]$

$\mathcal{P}$ has one rule of inference:

- Modus Ponens (MP): From $\mathbf{A}$ and $\mathbf{A} \supset \mathbf{B}$, infer $\mathbf{B}$.

A proof of a wff $\mathbf{A}$ is a sequence of wffs $\mathbf{A}_1, \ldots, \mathbf{A}_n$ such that $\mathbf{A}_n$ is $\mathbf{A}$, and for all $k \in [n]$, at least one of the following holds:

- $\mathbf{A}_k$ is an axiom.
- $\mathbf{A}_k$ is inferred by MP from wffs $\mathbf{A}_i, \mathbf{A}_j$, where $i < j < k$.

A theorem is a wff that has a proof. If $\mathbf{A}$ is a theorem, we write $\vdash_{\mathcal{P}} \mathbf{A}$.

Observe that a "proof" is a purely syntactic concept. Wffs are just strings of symbols; we have not given them any meaning. Proofs say nothing about meaning – they just show that with a bunch of symbol manipulation, you can work the axioms into a theorem. Proofchecking can be performed simply by a machine.

A proof of $p \vee \sim p$:

(.1)	$\vdash [p \vee p] \supset p$	Axiom 1
(.2)	$\vdash [[p \vee p] \supset p] \supset [[\sim p \vee [p \vee p]] \supset [p \vee \sim p]]$	Axiom 3
(.3)	$\vdash [\sim p \vee [p \vee p]] \supset [p \vee \sim p]]$	MP: .1, .2
(.4)	$\vdash [p \supset [p \vee p]] \supset [p \vee \sim p]]$	Abbr
(.5)	$\vdash p \supset [p \vee p]$	Axiom 2
(.6)	$\vdash p \vee \sim p$	MP: .4, .5

Now we will assign meaning to wffs, and try to determine if there is anything special about those wffs that are theorems of $\mathcal{P}$. Perhaps even all wffs are theorems of $\mathcal{P}$ – that would make $\mathcal{P}$ fairly worthless.

Note that in general it would not be useful to declare all wffs true or false. The wff p is neither true or false – it depends on the value of p. Hence, we define the meaning of a particular wff with respect to some context.

An assignment is a function from the set of propositional variables to the set of truth values $\{\mathsf{T}, \mathsf{F}\}$. We then define the value $\mathcal{V}_\varphi \mathbf{A}$ of a wff $\mathbf{A}$ with respect to an assignment φ recursively by:

- $\mathcal{V}_\varphi \mathbf{p} = \varphi \mathbf{p}$
- $\mathcal{V}_\varphi \sim \mathbf{A} = \mathsf{T}$ iff $\mathcal{V}_\varphi \mathbf{A} = \mathsf{F}$
- $\mathcal{V}_\varphi [\mathbf{A} \vee \mathbf{B}] = \mathsf{T}$ iff $\mathcal{V}_\varphi \mathbf{A} = \mathsf{T}$ or $\mathcal{V}_\varphi \mathbf{B} = \mathsf{T}$

We say $\mathbf{A}$ is a tautology or valid, and write $\models_{\mathcal{P}} \mathbf{A}$ if $\mathcal{V}_\varphi \mathbf{A} = \mathsf{T}$ for all assignments φ. Tautologies are in a sense, the "true" statements of $\mathcal{P}$; they are true under any assignment.

We'll now attempt to characterize the theorems of $\mathcal{P}$. One would hope that $\vdash p \vee \sim p$, but $\nvdash p \wedge \sim p$. More precisely, we would like all theorems of $\mathcal{P}$ to be valid – $\mathcal{P}$ would not be very useful if it could prove invalid wffs.
It turns out that all theorems of $\mathcal{P}$ are valid. That property is called soundness, and we say $\mathcal{P}$ is sound.

Theorem (Soundness). $\vdash_{\mathcal{P}} \mathbf{A} \implies \vDash_{\mathcal{P}} \mathbf{A}$

Proof. By induction. Note that to prove some property holds for all theorems of $\mathcal{P}$, it suffices to show the property holds for all axioms, and the property is preserved by all rules of inference. It's easily seen that all three axiom schemas are valid. Suppose then $\mathbf{A}$ and $[\mathbf{A} \supset \mathbf{B}]$ are valid. Then, for any assignment φ, $\mathcal{V}_\varphi \mathbf{A} = \mathsf{T}$ and $\mathcal{V}_\varphi [\mathbf{A} \supset \mathbf{B}] = \mathsf{T}$, so $\mathcal{V}_\varphi \mathbf{B} = \mathsf{T}$. Hence, $\mathbf{B}$ is valid. By induction, all theorems are valid. $\square$

We now know that all of the theorems of $\mathcal{P}$ are valid. The next natural question to ask is if $\mathcal{P}$ can actually prove all of those wffs that are valid.

Again, it turns out the answer is yes. That property is called completeness, and we say $\mathcal{P}$ is complete.

Theorem (Completeness). $\vDash_{\mathcal{P}} \mathbf{A} \implies \vdash_{\mathcal{P}} \mathbf{A}$

Consider the set $\mathrm{Thm}(\mathcal{P}) = \{\mathbf{A} : \vdash_{\mathcal{P}} \mathbf{A}\}$. Is it decidable?

Consider the set $\mathrm{Thm}(\mathcal{P}) = \{\mathbf{A} : \vdash_{\mathcal{P}} \mathbf{A}\}$. Is it decidable?
Yes. By soundness and completeness, $\vdash_{\mathcal{P}} \mathbf{A} \iff \vDash_{\mathcal{P}} \mathbf{A}$. Given $\mathbf{A}$, we can determine if $\vDash \mathbf{A}$ by checking all possible assignments to the variables that occur in $\mathbf{A}$. Since $\mathbf{A}$ is a finite string, only finitely many variables occur in it.

The system $\mathcal{P}$ of propositional calculus is an example of a formal system. A formal system $\mathcal{A}$ consists of a language of wffs together with a set of axioms and rules of inference.
A proof of a wff **A** in $\mathcal{A}$ is a list of wffs, the last of which is **A**, and each of which is either an axiom or follows from preceding wffs by a rule of inference. A wff **A** that has a proof is called a theorem of $\mathcal{A}$, and we write $\vdash_{\mathcal{A}} \mathbf{A}$.

One can then define some natural evaluation system on wffs. If a wff **A** is true under the particular evaluation, one says the wff is valid, and writes $\models_{\mathcal{A}} \mathbf{A}$. The exact meaning of validity depends on the particular evaluation chosen.

- $\mathcal{A}$ is inconsistent iff $\mathcal{A}$ proves a contradiction.
- $\mathcal{A}$ is sound iff $\vdash_{\mathcal{A}} \mathbf{A} \implies \models_{\mathcal{A}} \mathbf{A}$.
- $\mathcal{A}$ is complete iff $\models_{\mathcal{A}} \mathbf{A} \implies \vdash_{\mathcal{A}} \mathbf{A}$.

$\mathcal{A}$ is decidable iff $\mathrm{Thm}(\mathcal{A}) = \{\mathbf{A} : \vdash_{\mathcal{A}} \mathbf{A}\}$ is decidable.

The system $\mathcal{P}$ shows how one can combine statements using propositional connectives. But $\mathcal{P}$ is not powerful enough to express even simple mathematical statements such as,

$$\text{There is a prime between } n \text{ and } 2n, \text{ for } n > 1$$

We need to somehow strengthen our language. Statements can be regarded as asserting individuals have certain properties, or are related to each other in certain ways. The key concept to add is the ability to express that properties hold for all, or for some individuals in our domain of discourse. To accomplish that, we'll add quantifiers and individuals to the system. The stronger system is called first-order logic, or predicate logic.

The alphabet of $\mathcal{F}$ consists of:

- The symbols $[,], \sim, \vee, \forall$
- Individual variables: $u, v, w, x, y, z, u_1, v_1, \ldots$
- n-ary function variables: $f^n, g^n, h^n, \ldots$ for each $n \geq 1$.
- Propositional variables: $p, q, r, s, \ldots$
- n-ary predicate variables: $P^n, Q^n, R^n, \ldots$ for each $n \geq 1$.

Some variables may be designated as constants. We will not allow quantification over constants.

The terms and wffs of $\mathcal{F}$ are defined inductively:

- If $\mathbf{x}$ is an individual variable, $\mathbf{x}$ is a term.
- If $\mathbf{t}_1, \ldots, \mathbf{t}_n$ are terms and $\mathbf{f}^n$ is a function variable, $\mathbf{f}^n\mathbf{t}_1 \ldots \mathbf{t}_n$ is a term.
- If $\mathbf{t}_1, \ldots, \mathbf{t}_n$ are terms and $\mathbf{P}^n$ is a predicate variable, $\mathbf{P}^n\mathbf{t}_1 \ldots \mathbf{t}_n$ is a wff. Also, propositional variables are wffs.
- If $\mathbf{A}$ is a wff, then $\sim \mathbf{A}$ is a wff.
- If $\mathbf{A}, \mathbf{B}$ are wffs, then $[\mathbf{A} \vee \mathbf{B}]$ is a wff.
- If $\mathbf{A}$ is a wff and $\mathbf{x}$ is an individual variable, then $\forall \mathbf{x}\mathbf{A}$ is a wff.

Note that quantification is permitted only over individual variables.

It's helpful to (again) introduce some abbreviations in the meta-language. We'll adopt the same abbreviations as for $\mathcal{P}$. Additionally,

- $\exists \mathbf{x}\mathbf{A}$ stands for $\sim \forall \mathbf{x} \sim \mathbf{A}$

Note that we can think of unary predicate variables as sets. Associated with any set S, is the property P of being in that set, and associated with any property P is the set S of elements with that property. Then, $x \in S$ is equivalent with Px. Hence, for convenience we will identify sets with unary predicates. If $\mathbb{N}$ were a constant denoting the natural numbers and 0 a constant denoting zero, one would express $0 \in \mathbb{N}$ in the system as $\mathbb{N}0$.

Occurences of variables in wffs can either be “bound” or “free”. An occurence of a variable is bound in a wff iff it occurs within the scope of a universal quantifier that quantifies it. Otherwise, it is free.

Which occurences are bound/free?

- $Py \vee \forall x Px$
- $Px \vee \forall x \forall x Px$

If **A** is a wff, **x** is an individual variable, and **t** is a term, then $\mathbf{A}\{\mathbf{t}/\mathbf{x}\}$ denotes the formula that results if all free occurences of **x** in **A** are substituted with **t**. E.g.,

$$Px \vee \forall x Qx\{y/x\} = Py \vee \forall x Qx$$

The axioms of $\mathcal{F}$ consist of all wffs of the following form:

1. $[\mathbf{A} \vee \mathbf{A}] \supset \mathbf{A}$
2. $\mathbf{A} \supset [\mathbf{B} \vee \mathbf{A}]$
3. $[\mathbf{A} \supset \mathbf{B}] \supset [[\mathbf{C} \vee \mathbf{A}] \supset [\mathbf{B} \vee \mathbf{C}]]$
4. $\forall \mathbf{x}\mathbf{A} \supset \mathbf{A}\{\mathbf{t}/\mathbf{x}\}$, where $\mathbf{t}$ is a term[1]
5. $\forall \mathbf{x}[\mathbf{A} \vee \mathbf{B}] \supset [\mathbf{A} \vee \forall \mathbf{x}\mathbf{B}]$ where $\mathbf{x}$ is not free in $\mathbf{A}$.

$\mathcal{F}$ has two rules of inference.

- Modus Ponens (MP): From $\mathbf{A}$ and $\mathbf{A} \supset \mathbf{B}$, infer $\mathbf{B}$.
- Universal Generalization (Gen): From $\mathbf{A}$ infer $\forall \mathbf{x}\mathbf{A}$, where $\mathbf{x}$ is an individual variable.

[1] Some restrictions apply

An interpretation $\mathcal{M} = \langle \mathcal{D}, \mathcal{J} \rangle$ of $\mathcal{F}$ is a non-empty set $\mathcal{D}$, called the universe or domain of individuals and a mapping $\mathcal{J}$ defined on the constants of $\mathcal{F}$.
An assignment into $\mathcal{M}$ is a function φ defined on the variables of $\mathcal{F}$ satisfying,

- If $\mathbf{x}$ is an individual variable, then $\varphi\mathbf{x}$ is an element of $\mathcal{D}$.
- If $\mathbf{f}^n$ is an n-ary function variable, then $\varphi\mathbf{f}^n$ is a function from $\mathcal{D}^n$ to $\mathcal{D}$.
- If $\mathbf{p}$ is a propositional variable, then $\varphi\mathbf{p}$ is a truth value.
- If $\mathbf{P}^n$ is an n-ary predicate variable, then $\varphi\mathbf{P}^n$ is a function from $\mathcal{D}^n$ to $\{\mathsf{T}, \mathsf{F}\}$.

$\mathcal{J}$ should satisfy the same properties for the constants of $\mathcal{F}$.

Given an interpretation $\mathcal{M}$ and assignment φ into $\mathcal{M}$, one can in a straightforward way define $\mathcal{V}_{\varphi}^{\mathcal{M}}\mathbf{A}$, the value of $\mathbf{A}$ with respect to φ in $\mathcal{M}$.

We say $\mathbf{A}$ is valid in $\mathcal{M}$, and write $\mathcal{M} \models_{\mathcal{F}} \mathbf{A}$ iff $\mathcal{V}_{\varphi}^{\mathcal{M}}\mathbf{A} = \mathsf{T}$ for all assignments φ.

We say $\mathbf{A}$ is valid, and write $\models_{\mathcal{F}} \mathbf{A}$ iff $\mathbf{A}$ is valid in every interpretation.

Theorem. $\mathcal{F}$ is both sound and complete. $\vdash_{\mathcal{F}} \mathbf{A} \iff \models_{\mathcal{F}} \mathbf{A}$.

The completeness of $\mathcal{F}$ was proved by Gödel in 1930, and is known as Gödel's completeness theorem.

Valid?

- $[\exists x Px \wedge \exists x Qx] \supset \exists x[Px \wedge Qx]$
- $\exists x[Px \wedge Qx] \supset [\exists x Px \wedge \exists x Qx]$
- $\exists x Px \supset \forall x Px$
- $\forall x Px \supset \exists x Px$

$\mathcal{F}$ can be used as a basis to formalize many branches of mathematics. For example, one might add to $\mathcal{F}$ an individual constant 0, a function constant $+$ and binary predicate constant $=$, along with the following axioms. We'll write write $+$ and $=$ in infix position, using $\mathbf{t}_1 + \mathbf{t}_2$ instead of $+\mathbf{t}_1\mathbf{t}_2$.

1. $\forall x\ x = x$
2. $\forall x \forall y \forall z [x = y \wedge y = z \supset x = z]$
3. $\forall x \forall y \forall z [x = y \supset x + z = y + z]$
4. $\forall x \forall y \forall z [x + (y + z) = (x + y) + z]$
5. $\forall x [x + 0 = x]$
6. $\forall x \exists y [x + y = 0]$
7. $\forall x \forall y [x + y = y + x]$

One thus obtains the theory of additive abelian groups. In a similar manner, one can formalize a variety of mathematical disciplines.

Another theory that can be defined is Peano Arithmetic (PA). PA adds the basic axioms of the natural numbers.

In the 1920's, David Hilbert proposed the following program for the advancement of mathematics.

1. Define a formal system $\mathcal{A}$ capable of expressing the all of the ideas in mathematics.
2. Prove the consistency of $\mathcal{A}$, using only simple "finitistic" methods whose validity could not be doubted.
3. Prove the completeness of $\mathcal{A}$. Specifically, show $\mathcal{A}$ can prove all true statements in mathematics.
4. Solve the decision problem for $\mathcal{A}$.

Upon completion, the decision algorithm for $\mathcal{A}$ would be an algorithm that could determine whether an arbitrary mathematical statement is true. Mathematics would be reduced to finding interesting things to ask questions about, and improving the efficiency of the decision algorithm. Humans would never need to prove theorems again.

In 1931, Gödel showed that not a single goal of the program was achievable. In fact, any formal system whose proofs can be checked and is powerful enough to express elementary number theory cannot be both consistent and complete.

We say $\mathcal{A}$ is recursively axiomatized if the set of axioms of $\mathcal{A}$ is decidable.

Theorem (Gödel, 1931). Let $\mathcal{A}$ be any recursively axiomatized extension of PA. If $\mathcal{A}$ is consistent, then $\mathcal{A}$ is incomplete.

Note the restriction to recursively axiomatized systems is important – otherwise, one could simply consider the system whose axioms are the set of true statements. Such a system defeats the purpose of having axiomatic reasoning, as then there is no way to determine if a given formula is an axiom.

We'll prove a similar theorem that illustrates the idea of Gödel's theorem.

Theorem. Let $\mathcal{A}$ be any recursively axiomatized extension of PA. If $\mathcal{A}$ is sound, then $\mathcal{A}$ is incomplete. That is, $\mathcal{A}$ cannot be both sound and complete.

Proof. There are two key ideas.

- First, PA is powerful enough to express statements about programs. Specifically, for any program `P`, one can represent the statement that "`P(P)` halts" in PA, and hence in $\mathcal{A}$.
- Second, since $\mathcal{A}$ is recursively axiomatized, one can write a proof-checker program `PROOF` such that `PROOF(A, P)` returns true if `P` is a proof of `A`, and false otherwise.

Consider the program:

```
CONFUSE(P):
   For n = 1 to infinity:
      If PROOF("P(P) halts", n)
         Loop Forever
      If PROOF("P(P) does not halt", n)
         Halt
```

That is, given `P`, `CONFUSE` simultaneously searches for proofs of the statements "`P(P)` halts" and "`P(P)` doesn't halt".
Now assume $\mathcal{A}$ is sound and consider `CONFUSE(CONFUSE)`.

```
CONFUSE(CONFUSE):
   For n = 1 to infinity:
      If PROOF("CONFUSE(CONFUSE) halts", n)
         Loop Forever
      If PROOF("CONFUSE(CONFUSE) does not halt", n)
         Halt
```

- If $\mathcal{A}$ proves `CONFUSE(CONFUSE)` halts, then it does not halt.
- If $\mathcal{A}$ proves `CONFUSE(CONFUSE)` does not halt, then it halts.

Hence, if $\mathcal{A}$ proves either statement, $\mathcal{A}$ is unsound, so $\mathcal{A}$ proves neither statement. But then `CONFUSE(CONFUSE)` never finds either proof, and so it never halts. Hence, the statement "`CONFUSE(CONFUSE)` does not halt" is a true statement, but $\mathcal{A}$ can't prove it. $\mathcal{A}$ is incomplete.

One can never hope to formalize all of mathematics within a formal system that is sound and complete.
Soundness is important – a system isn't very useful if it proves false statements. Hence, soundness is the clear choice. But this means there will always be true statements the system cannot prove.
Most mathematicians don't run into these problems of unprovable statements. But every once in awhile they do appear. Examples include the Axiom of Choice and the Continuum Hypothesis, neither of which can be proved or refuted by ZF set theory.

If you're interested in studying logistic systems in much greater detail, take 21-600 and 21-700 with Peter Andrews.
The systems described here are those in his book *An Introduction to Mathematical Logic and Type Theory: To Truth Through Proof*.