Gödel's Turing: The Search for "Absolute" Epistemological Notions

Juliette Kennedy
Department of Mathematics and Statistics
University of Helsinki

I. In thinking about the origin of computability we are thinking about the question…

By which criteria do we shape, commit ourselves to, or otherwise assess standards of adequacy? This is the problem of *faithfulness*; the problem of what is lost whenever an intuitively given mathematical concept is made exact, or, beyond that, formalized; the problem, in words, of the *adequacy* of our mathematical definitions. It is a philosophical problem rather than a mathematical one, as the idea of "fit" cannot be subject to mathematical proof.

One coping mechanism: "thesis" language

- Church-Turing Thesis: equates the class of intuitively computable number-theoretic functions with the class of functions computable by a Turing Machine. (In its present formulation.)
- Weierstrass thesis: the $\varepsilon - \delta$ definition of continuity correctly and uniquely expresses the informal concept.
- Dedekind's thesis: the collection of Dedekind cuts gives the correct definition of the concept "line without gaps". (Alternatively, that Dedekind gave the correct definition of a natural number.)
- Area thesis: the Riemann integral correctly captures the idea of the area bounded by a curve.
- Hilbert's Thesis: "the steps of any mathematical argument can be given in a first order language (with identity)." WHO?

"There are theses everywhere."

Shapiro, "The open texture of computability." In *Computability—Turing, Gödel, Church, and beyond.*

Gödel viewed Turing's analysis of computability as paradigmatic, and the impact on his thinking, in 1946 and subsequently, was substantial: mathematically, Gödel's "transfer" of the Turing analysis of computability to the case of provability led to the first formulation of what has come to be known as Gödel's program for large cardinals. In the case of definability this transfer led to the fruitful concept of ordinal definability in set theory.

Philosophically: goal is decidability

Much of Gödel's philosophical work was directed toward the formulation of a view from which the unrestricted application of the Law of Excluded Middle to the entire cumulative hierarchy of sets could be justified.

Here, Gödel's *appropriation* of the Turing analysis lent power and plausibility to his search for a logically autonomous perspective, allowing an overview of logical frameworks, while not being entangled in any particular one of them—*for that is what absolute decidability entails*. It is a use of logical formalisms that is *localized* and *opportunistic* rather than global and foundational.

As with the notion of "finite set," one cannot think about computability, in its historical context or otherwise, without coming up against the intriguing conceptual anomalies, circularities and the like, which plagued the early attempts to ground the notion of effectivity, and plague them still. On the one hand, computation can be seen as a form of deduction, while on the other hand deduction is easily seen as a form of (non-deterministic) computation.

There is also the spectre of deviant encodings, which means that there is no principled way to recognize—in an absolute sense—the legitimacy of one's computational model. And while I will focus on Gödel's place in these developments, the difficult question how to rule out deviant encodings is part of a more general point: semantic concepts cannot be eliminated from our meta-mathematical discourse—even in the case of the concept "mechanical procedure."

"…demand that the semantic relation between numerals and numbers itself be intuitively computable."

Where do you analyze the notion of intuitively computable invoked above?

II. Computability in the 1930s

Gödel employed the class of *recursive* functions in his landmark 1931 paper.

Recursion was already known: Dedekind (1888, thm 126); Ackermann (1928) separated recursion from primitive recursion*, Rosza Péter simplified Ackermann's presentation (1932), coined the term "primitive recursive".

*Recursion is a much more general notion of computability than primitive recursion, allowing double recursions, partial functions, etc.

Church introduces the λ-calculus in 1932

Church developed the λ-calculus together with Kleene, a type-free and indeed, in Gandy's words, "logic free" model of effective computability, based on the primitives "function" and "iteration."

The phrase "logic free" is applicable only from the point of view of the later 1936 presentation of it, as Church's original presentation of the λ-calculus in his 1932* embeds those primitives in a "deductive formalism," in Hilbert and Bernays's terminology.

Computability in a logic

The attitude in Princeton initially (in the early 1930s) was that computability should be understood in terms of *calculability in a logic*.

(I mainly rely on the accounts given in Gandy's 1988 "The confluence of ideas in 1936" and Sieg's 1997 "Step by Recursive Step: Church's Analysis of Effective Calculability".)

"Computable in a logic" means…

"And let us call a function F of one positive integer *calculable within the logic* if there exists an expression f in the logic such that $f(\mu) = v$ is a theorem when and only when $F(m) = n$ is true, μ and v being the expressions which stand for the positive integers m and n." (Church, 1936)

Circularity?

Such functions F are recursive, according to Church, *if it is assumed that the logic's theorem predicate is recursively enumerable*.

Problem

Church's original presentation of the λ-calculus was found by his students Kleene and Rosser to be inconsistent.

Kleene, Rosser, "The inconsistency of certain formal logics", *Bulletin of the American Mathematical Society*, vol. 41 (1935)

"Not exactly what one dreams of having one's graduate students accomplish for one."

---Martin Davis

Kleene: History of computability in the period 1931-1933

"When it began to appear that the full system is inconsistent, Church spoke out on the significance of λ-definability, abstracted from any formal system of logic, as a notion of number theory."

Church's Thesis originates in 1933-4

CT begins with Church's verbal suggestions in 1933-4, to identify the λ-definable functions with the *effectively computable** ones.

(Following the literature, "effectively computable" is used here to mean "intuitively computable.")

Evidence for CT

Any function that *appeared* to be effectively computable, *was* λ-definable, and conversely.

(Much labor went into showing this!)

Gödel unconvinced…

Gödel: “Thoroughly unsatisfactory.”

Church reports the remark in a letter to Kleene dated November 29, 1935.

Another line of thought: Herbrand-Gödel Recursion

By 1934, compelled to "make the incompleteness results less dependent on particular formalisms," Gödel introduced in his Princeton lectures of spring 1934, the general recursive or Herbrand-Gödel recursive functions, as they came to be known, defining (along the way) the notion of *"formal system" as consisting of "symbols and mechanical rules relating to them."* Inference and axiomhood were to be witnessed by a finite (primitive recursive) procedure, also the syntax.

Formal system: symbols and mechanical rules

"We require that the rules of inference, and the definitions of meaningful formulas and axioms, be constructive; that is, for each rule of inference there shall be a finite procedure for determining whether a given formula B is an immediate consequence (by that rule) of given formulas A1,...An, and there shall be a finite procedure for determining whether a given formula A is a meaningful formula or an axiom. "

HG calculus continued

The calculus admits forms of recursions that go beyond primitive recursion. Roughly speaking, while primitive recursion is based on the successor function, in the Herbrand-Gödel equational calculus one is allowed to substitute other recursive functions in the equations, as long as this defines a unique function. (For example f(n) = f(n+1) does not define a unique function.)

The adequacy of the HG equational calculus

Gödel, letter to Martin Davis February 15, 1965:

"...I was, at the time of these lectures, not at all convinced that my concept of recursion comprises all possible recursions."

Looking at the system, there *is* little reason to suppose this.

CT crystallizes in 1935

In Church's lecture on what came to be known as "Church's Thesis" to the American Mathematical Society in 1935, Church uses the Herbrand-Gödel equational calculus as a model of effective computation, i.e. recursiveness in the "new sense," rather than the λ-calculus.

This was preceded by a complex development in which the functions defined in the HG-equational calculus were shown to be the same as the λ-definable functions.

These equivalences must be shown of the encodings of the various systems in some axiomatic system by means of the Kleene T-predicate.

Two approaches in 1936

In fact Church presented two approaches to computability in the AMS lectures and in his subsequent [1936], based on the lectures: Firstly algorithmic (what Gandy had called "logic-free"), based on what is now known as the untyped λ-calculus, i.e. the evaluation of the value f(m) of a function by the step-by-step application of an algorithm—and secondly logical, based on the idea of calculability in a logic.

Viewing computation as a form of deduction is very natural.

"My main point is this: computation is a special form of mathematical argument. One is given a set of instructions, and the steps in the computation are supposed to follow—follow deductively— from the instructions as given.

It is in this sense…that I am regarding computation as a special form of deduction, that I am saying I am advocating a logical orientation to the problem."

Kripke, "The Church-Turing "Thesis" as a special corollary of Gödel's completeness theorem." In *Computability: Gödel, Church, and Beyond.*

Hilbert's Thesis

Kripke used Hilbert's Thesis---any valid argument can be stated in a first order language---to show that the solution of the *Entscheidungsproblem* follows almost trivially from the Completeness Theorem for first order logic, together with the *Incompleteness* Theorem.

Entscheidungsproblem

Is there an algorithm to decide of a given sentence of first order logic, whether it follows from those axioms?

Asked by Hilbert at the ICM in Bologna in 1928, by Hilbert and Ackermann same year, etc.

Goes back to Hilbert's Tenth Problem, posed at the Paris ICM in 1900.

Kripke's proof

Suppose α is an algorithm for deciding whether a formula ϕ, given as input, is provable in first order logic or not. So α can decide any computation in the Kripke sense of computation, because any computation is of the form $\theta \models \psi$ (i.e. $\models \theta \rightarrow \psi$) for some first order θ and ψ. Let us take α^* to be an extension of α with some coding built in, e.g. enough arithmetic to get Gödel's Fixed Point Theorem. Let $\phi(x)$ be a supposed formula such that for every formula ψ of first order logic, since α decides whether ψ is valid or not, either $\phi(\text{“}\psi\text{”})$ or $\neg\phi(\text{“}\psi\text{”})$ is first order derivable from α^*. By Gödel's Fixed Point Lemma there is ψ such that $\alpha^* \models \psi \leftrightarrow \neg\phi(\text{“}\alpha^* \rightarrow \psi\text{”})$. Now

$[\alpha^* \models \psi]$ implies $[\alpha^* \models \phi(\text{“}\alpha^* \rightarrow \psi\text{”})]$ implies $[\alpha^* \models \neg\psi]$, a contradiction

$[\alpha^* \not\models \psi]$ implies $[\alpha^* \models \neg\phi(\text{“}\alpha^* \rightarrow \text{“}\psi\text{”})]$ implies $[\alpha^* \models \psi]$, a contradiction

Herbrand, 1929

The decision problem is "…the most important of those, which exist at present in mathematics."

Improvement: Hilbert-Bernays' 1939 Grundlagen der Mathematik II

HB present a logical calculus rather than a system of the type of Gödel's [1934]. The essential requirement here is that the proof predicate of the logic is *primitive recursive*.

This effects a precise gain: one reduces effectivity now to *primitive* recursion.

The problem with a logical approach

If effectivity is explained via a logic which is supposed to be given effectively, one must then introduce a new logic, by means of which the effectivity of the initial logic is to be analyzed.

It is natural to assume of the new logic, that it *also* should be given effectively. But then one must introduce a third logic by means of which this effectivity is to be analyzed. And so forth…

Shift in perspective had begun to set in anyway in 1934

Gandy: ". . . in 1934 the interest of the group shifted from systems of logic to the λ-calculus and certain mild extensions of it…"

Indeed the Herbrand-equational calculus is not a system of logic per se. Nor is Kleene's 1936 system based on the concept of μ-recursion, a logical calculus; and nor is Post's model of computability presented (also) in 1936 (based on work he had done in the 1920s).

All of these are conceptions of computability given, primarily, mathematically—*but there was no reason whatsoever to believe in their adequacy.*

Confluence not enough

They had *confluence*, i.e. they knew that their various systems were equivalent in the sense of giving the same class of functions. But they lacked a grounding example.

Turing 1936

Rather than calculability in a logic, Turing analyzes effectivity in terms of an informal, fully sharpened, mathematical notion: the concept of a Turing machine.

Turing also solves the *Entscheidungsproblem*!

First proves the unsolvability of the halting problem: given any Turing Machine, does it halt?

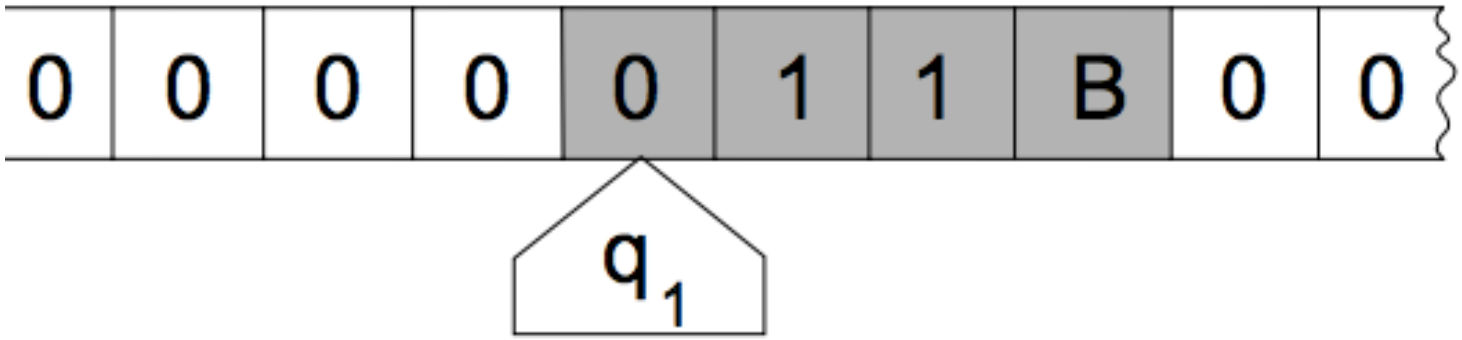
Then shows that provability in FOL is decidable iff the halting problem is solvable.

Church had solved the *Entscheidingsproblem* shortly before…

Turing's Machines

Turing's machine model of computability consists of a tape scanned by a reader, together with a set of simple instructions in the form of quintples.

The analysis consisted of two elements: a conceptual analysis of human effective computation, together with a mathematical precisification of it consisting of rules given by the simple instructions "erase," "print 1," "move left," and "move right."


0 0 0 0 0 1 1 B 0 0
q1

Reactions

"Turing's computability is intrinsically persuasive but λ-definability is not intrinsically persuasive and

general recursiveness scarcely so (its author Gödel being at the time not at all persuaded)."

Kleene, 1981, "The theory of recursive functions, approaching its centennial." *Bull. Amer. Math. Soc. (N.S.), 5(1)*.

.

Gödel to Wang

"The resulting definition of the concept of mechanical by the sharp concept of "performable by a Turing machine" is both correct and unique...Moreover it is absolutely impossible that anybody who understands the question and knows Turing's definition should decide for a different concept."

G to Wang continued

"The sharp concept is there all along, only we did not perceive it clearly at first. This is similar to our perception of an animal far away and then nearby. We had not perceived the sharp concept of mechanical procedure sharply before Turing, who brought us to the right perspective. "

Grounding

For the logicians of the time, then, the Turing Machine was not just another in the list of acceptable notions of computability—it was the *grounding* of all of them.

The Turing Machine is computability's beating heart.

Generality of the Incompleteness Theorems

A precise notion of formal system was needed for settling the question, taken up by Gödel himself in his 1931 paper on Incompleteness, whether those theorems are completely general, that is, whether they apply to any formal system containing arithmetic, and not just Principia and systems related to it. Gödel was careful to say at the end of his 1931 paper that this had not been shown. The issue lingered for some time after the Incompleteness Theorems had been published.

Turing analysis resolves the issue in Gödel's view

"In consequence of later advances, in particular of the fact that, due to A. M. Turing's work, a precise and unquestionably adequate definition of the general concept of formal system can now be given, the existence of undecidable arithmetical propositions and the non-demonstrability of the consistency of a system in the same system can now be proved rigorously for *every* consistent formal system containing a certain amount of finitary number theory." Gödel, 1965, Postscriptum to his 1934 lectures.

. . . Turing's work gives an analysis of the concept of "mechanical procedure" (alias algorithm or computation procedure or "finite combinatorial procedure"). This concept is shown to be equivalent with that of a "Turing machine." A formal system can simply be defined to be any mechanical procedure for producing formulas, called provable formulas. For any formal system in this sense there exists one in the [usual] sense that has the same provable formulas (and likewise vice versa). . .

"That my [incompleteness] results were valid for all possible formal systems began to be plausible for me (that is since 1935) only because of the Remark printed on p. 83 of 'The Undecidable' . . . But I was completely convinced only by Turing's paper. "

Gödel, letter to Kreisel (printed in Odifreddi.)

Gandy: Turing's isolation from the logical milieu of Princeton was the key

"It is almost true to say that Turing succeeded in his analysis because he was not familiar with the work of others. . . The bare hands, do-it-yourself approach does lead to clumsiness and error. But the way in which he uses concrete objects such as exercise book and printer's ink to illustrate and control the argument is typical of his insight and originality. Let us praise the uncluttered mind. "

"All the work described in Sections 14.3-14.977 [work of Church etc JK] was based on the mathematical and logical (and not on the computational) experience of the time. What Turing did, by his analysis of the processes and limitations of calculations of human beings, was to clear away, with a single stroke of his broom, this dependence on contemporary experience, and produce a characterization which—within clearly perceived limits—will stand for all time. "