Gödel on Computability

(to appear in: Philosophia Mathematica)

Wilfried Sieg

July 21, 2005

Technical Report No. CMU-PHIL-174

Philosophy

Methodology

Logic

Carnegie Mellon

Pittsburgh, Pennsylvania 15213

Gödel on computability*

Wilfried Sieg
Carnegie Mellon University

Around 1950, both Gödel and Turing wrote papers for broader audiences.[1] Gödel drew in his *1951* dramatic philosophical conclusions from the general formulation of his second incompleteness theorem. These conclusions concerned the nature of mathematics and the human mind. The general formulation of the second theorem was explicitly based on Turing's 1936 reduction of finite procedures to machine computations. Turing gave in his *1954* an understated analysis of finite procedures in terms of Post production systems. This analysis, prima facie quite different from that given in 1936, served as the basis for an exposition of various unsolvable problems. Turing had addressed issues of mentality and intelligence in contemporaneous essays, the best known of which is of course *Computing machinery and intelligence.* Gödel's and Turing's considerations from this period intersect through their attempt, on the one hand, to analyze finite, mechanical procedures and, on the other hand, to approach mental phenomena in a scientific way.

Neuroscience or brain science was an important component of the latter for both: Gödel's remarks in the Gibbs Lecture as well as in his later conversations with Wang and Turing's *Intelligent Machinery* can serve as clear evidence for that.[2] Both men were convinced that some mental processes are not mechanical, in the sense that Turing machines cannot mimic them. For Gödel, such processes were to be found in mathematical experience and he was led to the conclusion that mind is separate from matter. Turing simply noted that for a machine or a brain it is not enough to be converted into a universal (Turing) machine in order to become intelligent: "discipline", the characteristic

* Remarks and suggestions from Martin Davis, John Dawson, Solomon Feferman, Clark Glymour, Robert Hadley, Hourya Sinaceur, Guglielmo Tamburrini, Richard Tieszen, and Neil Tennant have improved this essay; my sincere thanks to all.

[1] I am in particular referring to *Gödel 1951* and *Turing 1954*, but also to *Gödel 1947* and *Turing 1950*.

[2] *Gödel 1951*, p. 312, *Wang 1974*, pp. 324-6, and *Turing 1948* pp.120-1.

feature of such a machine, has to be supplemented by "initiative", and it is a central scientific task "to discover the nature of this residue as it occurs in man, and to try and copy it in machines". (*Turing 1948*, p. 125)

In this essay I focus exclusively on Gödel's multifaceted work concerning the *analysis of mechanical procedures*, which he seems to have begun sometime around 1950. His considerations rely emphatically on Turing's paper *On computable numbers* and culminate in the enigmatic formulation of the 1964 Postscriptum to his Princeton Lectures where he writes, "Turing's work gives an analysis of the concept 'mechanical procedure'... This concept is shown to be equivalent with that of a 'Turing machine'." Gödel's analytic work must be seen in contrast with, or as a deepening of, his earlier attempts to explicate effectively calculable functions as general recursive ones.

Gödel introduced general recursive functions in his Princeton Lectures. Their definition is explicitly connected with the considerations of the Hilbert School on calculability and in particular with Herbrand's characterization of classes of *finitistically calculable functions*. His presentation is historically and conceptually intertwined also with Church's proposals to identify effective calculability first with λ-definability and then with general recursiveness. Only the recognition of the *absoluteness* of general recursive functions, however, convinced Gödel in 1936 that a stable and important class of functions had been found.[3] He emphasized this feature of general recursive functions in the strongest possible terms even ten years later, when he made his *Remarks before the Princeton bicentennial conference on problems in mathematics.*

It is deeply ironic that Gödel, in his Gibbs lecture, views the reduction of the concept of finite procedure to that of a finite machine as "the most satisfactory way" to arrive at a precise definition, whereas Turing, in his 1954 essay, does not even mention machines: his analysis is presented purely in terms

[3] This observation is formulated as a *Bemerkung* added in proof to *Gödel 1936*, and as a *Remark* in the paper's first translation that was published on p. 83 of *The Undecidable*, i.e., *Davis 1965*. The significance of this discovery was described by Gödel in a letter to Kreisel of 1 May 1968: "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." This quote is from *Odifreddi* (p. 65).

of combinatorial operations on finite discrete configurations, and his thesis states that all such combinatory processes can be reduced to Post canonical systems. I will explore how these crisscrossing attempts of characterizing mechanical procedures were motivated and how they unfolded in the two decades from 1934 to 1954. The unfolding in Gödel's case remained incomplete even during the following two decades. After all, Gödel's *1972* charged Turing with having committed a *philosophical error* in the very work that provided, according to Gödel's judgment in *1964*, the basis for "a precise and unquestionably adequate definition of the general notion of formal system".

I. Effective calculability[4]

The following is a brief reminder of the reasons why a rigorous mathematical formulation of this informally understood concept was crucial. For Gödel, it was essential to give a general definition of formal system and, thus, to have the means of formulating his incompleteness theorems in full generality for *all* formal systems (and not just for Principia Mathematica and related systems). Turing was fascinated by Hilbert's decision problem, the *Entscheidungsproblem* in the title of his 1936 paper, and suspected that it had a negative solution; but to give such a solution Turing had to characterize first the mechanical procedures involved in the very formulation of the problem. Church, finally, wanted to address the logical decision problem, but also more standard mathematical problems, for example in his *1936*, "the problem of topology, to find a complete set of calculable invariants of closed three-dimensional simplicial manifolds under homeomorphisms". Thus, we see in the work of these three pioneers the absolute need for a mathematically precise concept *corresponding* to the informal notion of effective calculation or mechanical procedure.

This correspondence has since been articulated as Church's or Turing's Thesis. In early 1934 Church had proposed in a conversation with Gödel to identify effective calculability of number-theoretic functions with their λ-

[4] This part of the history of computability has been widely discussed, for example, in *Gandy 1988* and *Soare 1999*; contributions to that discussion, with a distinctive perspective on Gödel's and Turing's impact, are found in my papers *1994*, *1997*, and *2005*.

definability. More than a year later he reported in a letter to Kleene on Gödel's reaction to his proposal:[5] Gödel viewed it as "thoroughly unsatisfactory" and made a counterproposal, namely, "to state a set of axioms which would embody the generally accepted properties of this notion [i.e., effective calculability], and to do something on that basis". However, instead of formulating axioms for that notion in his ongoing Princeton Lectures, Gödel modified Herbrand's characterization of classes of finitistically calculable functions and introduced the class of general recursive ones. The step from Herbrand's formulation to Gödel's is described in detail in my *2005*.

Gödel defined general recursive functions as those functions whose values can be computed in an equational calculus, using some recursion equations as starting points and extremely elementary substitution and replacement operations as inference rules. *Church 1936* and *Hilbert & Bernays 1939* considered more general calculi in which such computations can be carried out. They explicated the informal concept of effective calculability for number-theoretic functions by one core notion, namely, the formal computability of their values *in a logic* (as Church put it) or *in a deductive formalism* (as Hilbert & Bernays suggested). Gödel did *not* think of general recursiveness as a rigorous explication of effective calculability. What reads prima facie as a statement of Church's Thesis, namely footnote 3 of *Gödel 1934*, is not to be interpreted in that way at all. Gödel wrote to Martin Davis, "... it is not true that footnote 3 is a statement of Church's Thesis. The conjecture stated there only refers to the equivalence of 'finite (computation) procedure' and 'recursive procedure'. However, I was, at the time of these lectures, not at all convinced that my concept of recursion comprises all possible recursions; ..." What Gödel tried to do is to characterize the computations of values of recursively specified number-theoretic functions by uniform and arithmetically meaningful steps.

Indeed, for Gödel, the crucial point was the specification of systematic rules for carrying out computations or, to put it differently, for deriving equations. That point of view was also expressed by Kleene, who wrote with

[5] Church's letter to Kleene of 29 November 1935 was partially published in *Davis 1982*.

respect to the definition of general recursive functions that "it consists in specifying the form of equations and the nature of the steps admissible in the computations of the values, and in requiring that for each given set of arguments the computation yield a unique number as value". (*Kleene 1936*, p. 727) In a letter to van Heijenoort, dated 14 August 1964, Gödel asserted, "it was exactly by specifying the rules of computation that a mathematically workable and fruitful concept was obtained". The formulation of the equational calculus with purely formal, mechanical rules brought out clearly what Herbrand, according to Gödel, had failed to see, namely, "that the computation (for all computable functions) proceeds by exactly the same rules".

Church defended the identification of effective calculability with general recursiveness in his *1935* and tried to *prove* that every effectively calculable function is general recursive in *Church 1936*. By then, Church had significant additional information with which to support his proposal, namely, i) the quasi-empirical work of Kleene and Rosser showing that all known effectively calculable functions are indeed λ-definable, and ii) the mathematical equivalence of λ-definability with general recursiveness (but later also with μ-recursiveness that had been introduced in *Kleene 1936*). Church's attempt to prove that every effectively calculable function is general recursive was, however, semi-circular in the sense that he assumed without good reason that the necessarily elementary calculation steps have to be recursive.[6] (Adapting the ideas underlying the proof of Kleene's normal form theorem, Church's argument using this recursiveness assumption is perfectly direct; Gandy called it appropriately the "step-by-step argument".)

Hilbert & Bernays introduced in Supplement II of their *1939* the concept of a *reckonable function*. They explicitly imposed recursiveness conditions on deductive formalisms and proved that, if formalisms satisfy these conditions, then they allow the calculation of the values for exactly the general recursive functions. Their procedure can be viewed as embedding Gödel's equational

[6] This semi-circularity and the proof based on Church's "central thesis" are described in my *1997*.

calculus into formalisms satisfying the recursiveness conditions. The crucial condition requires the formalisms' proof predicates to be primitive recursive. Their mathematically rigorous presentation supports the claim, relative to formalisms satisfying the recursiveness conditions, that the general recursive functions are *absolute* in the sense in which Gödel formulated matters at the Princeton meeting of 1946; that sense is discussed in the next section.

II. Absoluteness

The Princeton meeting is of interest not only because of Gödel's contribution, but also because of Tarski's talk that was focused on the decision problem.[7] Tarski's choice of the decision problem as the main concern for the conference was motivated, as he put it, historically, materially, and heuristically: historically, as he connected it directly to Hilbert's concerns in *1900*; materially, as he thought "that the task of logic is to mechanize thinking"; heuristically, as "many diverse problems can be couched in terms of it". The decision problem raised for Tarski the intuitive question, is there "a mechanical means of deciding whether any given statement of a formal system is a theorem". It appeared to him that "the precise instrument for treating the decision problem is the Gödel-Herbrand notion of a general recursive function", and he argued that the problem's precise formulation coincides with the mathematical question, whether the set of theorems of a formal theory is recursive. In his argument Tarski used as an intermediate step the unsupported claim that the theorems of any formal system are recursively enumerable, as Church had done in 1936. He admitted that some logicians had doubts as to the adequacy of general recursiveness "to handle the intuitive content of the decision problem", yet he refrained from further remarks, since another speaker was to address the issue. It is likely that he had in mind Kleene or Gödel. As we will see, Gödel did refer to Tarski's talk and

[7] The following remarks are all based on the texts presented in *Sinaceur 2000*. Of interest in this context is also Tarski's letter to Gödel of 10 December 1946, responding to a letter from Gödel in which the topic of Gödel's Princeton talk had been described. (These letters were exchanged before the meeting.)

asserted that the absoluteness of general recursiveness makes that notion so significant.

In 1936 Gödel had already made an absoluteness claim for higher-type extensions of number theory in the remark added to his brief note *On the length of proofs*; the system S_1 is classical arithmetic, whereas for i>1 the S_i are the systems of arithmetic of order i. Here is the claim:

> It can, moreover, be shown that a function computable in one of the systems S_i, or even in a system of transfinite order, is computable already in S_1. Thus the notion 'computable' is in a certain sense 'absolute', while almost all metamathematical notions otherwise known (for example, provable, definable, and so on) quite essentially depend upon the system adopted. (*Collected Works I*, p. 399)

Ten years later Gödel reemphasized absoluteness, now understood in a more general way. In his contribution to the Princeton meeting, he considers not just higher-type extensions of arithmetic, but any formal theory that contains arithmetic. In a footnote added in 1965, Gödel makes that broadened absoluteness claim explicit:

> To be more precise: a function is computable in any formal system containing arithmetic if and only if it is computable in arithmetic, where a function f is called computable in S if there is in S a computable term representing f. (*Collected Works II*, p. 150)

In both formulations we have a clear meta-mathematical claim. As the earlier one pertains to particular, concretely presented formal theories, the step-by-step argument succeeds directly. The critical question is how the later claim can be proved, i.e., the assertion that functions computable in *any formal system* (containing arithmetic) are general recursive. Obviously, one has to exploit in some way the formal character of the system, for example, by making the assumption of Church or by imposing the recursiveness conditions of Hilbert & Bernays. Either way of proceeding reveals the relativity of Gödel's absoluteness. This is particularly damaging to Gödel's primary motivation for obtaining a rigorous notion of effective calculability: after all, a characterization of formal theories was needed for the general formulation of his incompleteness theorems, but here he presupposes a mathematically sharpened concept of such theories.

It is important to note that Church and Gödel at this point had very similar methodological views. Church's perspective on absoluteness comes out, for example, in a fascinating letter he sent to the Polish logician Pepis on 8 June

1937. (The full text of the letter is found in my *1997*, pp. 175-6.) Pepis had informed Church that he was engaged in finding an effectively calculable function that is not recursive. Church articulated in his reply an openly skeptical attitude roughly as follows: If there were such a function *f*, then for any given *a* there must exist a *b* together with a mathematical proof of $f(a)=b$.[8] This proof in turn must be formalizable in an extension, *by new principles*, of Principia Mathematica or other systems of symbolic logic. Any such new principle would have to be of a very complicated, not general recursive, sort and should raise concerns about its effective applicability. In sum, the assumption that *f* is effectively calculable, but not recursive, would force its computability in a theory that is no longer recursively presentable. This is a fittingly coherent perspective and parallels Gödel's.

The efforts of explicating effective calculability via computability in appropriate calculi and establishing absoluteness claims do not overcome the stumbling block for a general analysis, i.e., they do not provide a systematic reason for insisting that the elementary steps in computations have to be recursive. As we will see, Turing's work addresses this stumbling block directly. At the Princeton meeting Gödel mentioned Turing's notion of machine computability; however, it is only mentioned alongside general recursiveness without any emphasis that it might have a special role. Gödel starts his lecture with, "Tarski has stressed in his lecture (and I think justly) the great importance of the concept of general recursiveness (or Turing's computability)." He continues,

> It seems to me that this importance is largely due to the fact that with this concept one has for the first time succeeded in giving an absolute definition of an interesting epistemological notion, i.e., one not depending on the formalism chosen. (*Collected Works II*, p.150)

The philosophical significance of the notion is thus formulated in terms of its absolute definition, which makes it independent from particular formalisms. It is

[8] This remark obviously reflects Church's strong belief that every effectively calculable function must be computable in a logic; he adds to it parenthetically, "at least if we are not agreed on this then our ideas of effective calculability are so different as to leave no common ground for discussion".

not pointed out, however, that the correctness of the absoluteness claim depends on the fact that one is dealing with appropriately restricted formalisms.

Only in the Gibbs Lecture does Turing's computability become publicly a focal point for Gödel's attempt to characterize effective calculability. There he explores the implications of the incompleteness theorems, not in their original formulation, but rather in a "much more satisfactory form" that is "due to the work of various mathematicians". He stresses, "The greatest improvement was made possible through the precise definition of the concept of finite procedure, which plays such a decisive role in these results." In a footnote he explains what he understands by a "finite procedure" and points to the equivalent concept of a "computable function of integers", i.e., a function f "whose definition makes it possible actually to compute $f(n)$ for each integer n". There are, Gödel points out, different ways of arriving at a precise definition of finite procedure, which all lead to exactly the same concept. However, and here is the brief substantive remark on Turing, "The most satisfactory way ... [of arriving at such a definition] is that of reducing the concept of finite procedure to that of a machine with a finite number of parts, as has been done by the British mathematician Turing." (*Collected Works III*, pp. 304-5)

III. Finite machines

Gödel does not expand, in the Gibbs Lecture, on the brief remark just quoted. In particular, he gives no hint of how reduction is to be understood. There is no explanation of why such a reduction is the most satisfactory way of getting to a precise definition or, for that matter, of why the concept of a machine with a finite number of parts is equivalent to that of a Turing machine.[9] At this point, it seems, the ultimate justification lies in the pure and perhaps rather crude fact that finite procedures can be reduced to computations of finite machines. It is most interesting to note that Gödel elucidated already in his *1933*, as others had done

[9] It was only Robin Gandy who analyzed the concept of machine computation in his *1980*, i.e., he attempted to characterize computations of "discrete mechanical devices", excluding explicitly analog machines. Cf. my *2002*.

before him, the mechanical feature of effective calculations by pointing to the possibility that machines carry them out. When insisting that the inference rules of precisely described proof methods have to be "purely formal", he explains:

[The inference rules] refer only to the outward structure of the formulas, not to their meaning, so that they could be applied by someone who knew nothing about mathematics, or by a machine. This has the consequence that there can never be any doubt as to what cases the rules of inference apply to, and thus the highest possible degree of exactness is obtained. (*Collected Works III*, p. 45)

In the logic lectures Gödel gave at Notre Dame during the spring term of 1939, he discussed Hilbert's logical decision problem in an informal way, after having put it into the historical context of Leibniz's "Calculemus".[10] However, before arguing that results of modern logic prevent the realization of Leibniz's project, Gödel asserts that the rules of logic can be applied in a "purely mechanical" way and that it is therefore possible "to construct a machine which would do the following thing":

The supposed machine is to have a crank and whenever you turn the crank once around the machine would write down a tautology of the calculus of predicates and it would write down every existing tautology ... if you turn the crank sufficiently often. So this machine would really replace thinking completely as far as deriving of formulas of the calculus of predicates is concerned. It would be a thinking machine in the literal sense of the word. For the calculus of propositions you can do even more. You could construct a machine in form of a typewriter such that if you type down a formula of the calculus of propositions then the machine would ring a bell [if the formula is a tautology] and if it is not it would not. You could do the same thing for the calculus of monadic predicates.

Having formulated these positive results, Gödel asserts, "it is impossible to construct a machine which would do the same thing for the whole calculus of predicates", and continues:

So here already one can prove that Leibnitzens [sic!] program of the "calculemus" cannot be carried through, i.e. one knows that the human mind will never be able to be replaced by a machine already for this comparatively simple question to decide whether a formula is a tautology or not.

I mention these matters only to indicate the fascination Gödel had with the concrete mechanical realization of logical procedures (and his penchant for overly dramatic formulations).

The manuscript *193?* was a draft of a lecture Gödel never delivered, and its content may have been alluded to by Tarski in his Princeton talk. Gödel

[10] This is *Gödel 1939*. As to the character of these lectures, see also *Dawson*, p. 135.

examines there general recursiveness and Turing computability in great detail, with a methodological perspective that seems to be different from that of the Gibbs Lecture. He gives first, on pp. 167-8, a perspicuous presentation of the equational calculus and claims outright that it provides "the correct definition of a computable function". Then he asserts, "That this really is the correct definition of mechanical computability was established beyond any doubt by Turing." How did Turing establish that fact? Here is Gödel's answer:

> He [Turing] has shown that the computable functions defined in this way [via the equational calculus] are exactly those for which you can construct a machine with a finite number of parts which will do the following thing. If you write down any number $n_1, \ldots n_r$ on a slip of paper and put the slip of paper into the machine and turn the crank, then after a finite number of turns the machine will stop and the value of the function for the argument $n_1, \ldots n_r$ will be printed on the paper. (*Collected Works III*, p. 168)

The implicit claim is obviously that all mechanical procedures must be executable by machines with a finite number of parts. As above, we can raise the question, why should such machines be Turing machines?

Gödel does not answer this question; but that is not the crucial issue for my purposes at this point. Rather, I intend to look at the structure of the argument for the claim, "That this really is the correct definition of mechanical computability was established beyond any doubt by Turing." A literal reading amounts to this: the characterization of the computations of number-theoretic functions via the equational calculus is taken by Gödel to provide "the correct definition of computable function". That the class of computable functions is co-extensional with that of *mechanically* computable ones is then guaranteed by (what Gödel calls) Turing's proof of the equivalence of general recursiveness and Turing computability.[11] As there is no reason given for the first step in this argument, it really is to be viewed as a direct appeal to Church's Thesis. The second important step is justified by "Turing's proof".

If we go beyond the literal reading and think of the argument in terms of Turing's actual analysis (described in greater detail in the next section), then we could arrive at two parallel interpretations.

---

[11] In Turing's *1936* general recursive functions are not mentioned. Turing established in an Appendix to his paper the equivalence of his notion with λ-definability. As Church and Kleene had already proved the equivalence of λ-definability and general recursiveness, "Turing's Theorem" is established.

I) Turing starts out in his *1936* with calculations carried out by a (human) computer and argues that they can be taken to involve only very elementary processes; these processes can be carried out by a Turing machine that operates on strings of symbols. Gödel, this interpretation would maintain, takes the computations in the equational calculus somehow as "canonical representations" of the computer's calculations.

II) Turing states that the machines that operate on strings are easily seen to be equivalent to machines that operate on individual symbols; Gödel appeals to "Turing's proof" that recursiveness and Turing computability are equivalent.

The central analytic claims are hidden in I) and are unargued for in Gödel's case, whereas II) consists of mathematical proofs in both cases. I cannot see at all, how Turing's reductive steps can be adapted to argue for Gödel's analytic claim. After all, Turing shifts from arithmetically meaningful steps to symbolic processes, which underlie them and satisfy restrictive boundedness and locality conditions. These conditions cannot be imposed on arithmetic steps and are not satisfied by computations in the equational calculus. I will come back to these issues below.

Notice that in Gödel's way of thinking about matters at this juncture, the mathematical theorem stating the equivalence of Turing computability and general recursiveness plays a pivotal role. Church in his 1937 review of Turing's paper sees the situation in a quite different light. He asserts that Turing computability has the advantage over general recursiveness and λ-definability of "making the identification with effectiveness in the ordinary (not explicitly defined) sense evident immediately - i.e. without the necessity of proving preliminary theorems". Church's more detailed argument starts out as follows:

> The author [Turing] proposes as a criterion that an infinite sequence of digits 0 and 1 be "computable" that it shall be possible to devise a computing machine, occupying a finite space and with working parts of finite size, which will write down the sequence to any desired number of terms if allowed to run for a sufficiently long time. As a matter of convenience, certain further restrictions are imposed on the character of the machine, but these are of such a nature as obviously to cause no loss of generality - in particular, a human calculator, provided with pencil and paper and explicit instructions, can be regarded as a kind of Turing machine.

Church then draws the conclusion, "It is thus immediately clear that computability, so defined, can be identified with ... the notion of effectiveness as it appears in certain mathematical problems ..." These are fascinating considerations, but put the cart before the horse or, rather, reverse the direction of Turing's analysis. Informal calculations are simply identified with computations of finite machines that have a finite number of working parts. Those machines, Church claims, are up to convenient restrictions really Turing machines. As a human calculator can be regarded as a Turing machine, Turing computability can consequently be identified with effectiveness.

In a deep sense neither Church nor Gödel recognized the genuinely distinctive character of Turing's analysis, i.e., the move from arithmetically motivated calculations to general symbolic processes that underlie them. Most importantly in the given intellectual context, these processes have to be carried out programmatically by human beings: the *Entscheidungsproblem* had to be solved by us in a mechanical way; it was the normative demand of radical intersubjectivity between humans that motivated the step from axiomatic to formal systems.[12] It is for this very reason that Turing most appropriately brings in human computers in a crucial way and exploits the limitations of their processing capacities, when proceeding mechanically.

IV. Mechanical computors

Turing's work on computability is considered further in other writings of Gödel, or perhaps it would be better to say that the work appears as a topic of aphoristic remarks. Indeed, there are only three remarks that were published during Gödel's lifetime: i) the Postscriptum to the 1931 incompleteness paper, ii) the Postscriptum to the 1934 Princeton Lecture Notes, and iii) the note *A philosophical error in Turing's work.* In addition, we now have access to an informative letter to Ernest Nagel, which Gödel wrote on 25 February 1957, but

[12] That aspect was particularly clearly expressed by Frege in the Preface to volume I of *Grundgesetze der Arithmetik* when he compares his presentation with that of Euclid; the relevant passage is found on pp. 117-8 of *Translations from the Philosophical Writings of Gottlob Frege*, Peter Geach and Max Black (eds.), Third Edition, Blackwell, 1980.

never mailed. Finally, Hao Wang reports in his book *From Mathematics to Philosophy* on Gödel's views; of particular and pertinent interest are pp. 81-95 (under the heading *Mechanical procedures*) and pp. 324-326 (under the heading *Gödel on minds and machines*). The latter section of the book is a brief report on the Gibbs Lecture and the note *A philosophical error in Turing's work*, both of which had not been published yet. The former section contains remarks of Gödel's on general recursive functions as well as a detailed discussion of *Turing 1936*; however, from Wang's text it is not at all clear whether that discussion reflects Gödel's understanding.[13]

In the letter to Nagel, Gödel points out that Turing and Post clarified the general concept of *formalism* and that this clarification made the general formulation of the first incompleteness theorem possible. The brief Postscriptum added to *1931* on 28 August 1963 emphasizes this point for both incompleteness theorems; here is the full text:

> 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 notion of formal system can now be given, a completely general version of Theorems VI and XI is now possible. That is, it can be proved rigorously that in *every* consistent formal system that contains a certain amount of finitary number theory there exist undecidable arithmetic propositions and that, moreover, the consistency of any such system cannot be proved in the system. (*Collected Works I*, p. 195)

In the more extended Postscriptum written for the 1934 Princeton Lectures and dated 3 June 1964, Gödel repeats the above remark almost literally, but then seems to state a good reason why Turing's work provides the basis for a "precise and unquestionably adequate definition of the general concept of formal system":

> 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".

In a footnote attached to the last sentence Gödel refers to *Turing 1936* and points to the section in which Turing discusses the adequacy of his machine concept; this

---

[13] There is one new consideration reported in Wang's book (pp. 84-85): the perception of concepts together with the claim that Turing "brought us to the right perspective" and helped us to perceive "the sharp concept of mechanical procedures". - Wang's *A Logical Journey*, MIT Press, 1996, discusses in Chapter 6 the issues surrounding minds and machines; that discussion throws some additional light on the presentation in *Wang 1974*.

is section 9, entitled *The extent of the computable numbers.*[14] Gödel emphasizes in the footnote that "previous equivalent definitions of computability", including general recursiveness and λ-definability, "are much less suitable for our purposes". But there is no further elucidation by Gödel of the special character of Turing computability in this analytic context. That is unfortunate, as one would like to know how the analysis proceeds and how the (analyzed) concept is shown to be equivalent to Turing computability. We could go back to the parallel interpretations of *193?* and see whether they provide a key to addressing the problems. The difficulties we encountered there are, however, compounded here by Gödel's concrete reference to Turing's paper. So one should examine what Turing actually argues for in that part of his paper. I have done this in other publications, so I'll give only a very condensed version here.

Call a human computing agent who proceeds mechanically a *computor*; such a computor operates on finite configurations of symbols and, for Turing, deterministically so. The computer in Turing's paper is such a computor; computers in the modern sense are called machines. So the issue here is, how do we step from calculations of computors to computations of Turing machines? Turing explores "the extent of the computable numbers" (or, equivalently, of the effectively calculable functions) by considering two-dimensional calculations "in a child's arithmetic book". Such calculations are reduced to symbolic steps on linear configurations of such a simple character that they can be carried out by a Turing machine that operates on strings (instead of just letters). Indeed, Turing's argument concludes on pp. 137-8 as follows: "We may now construct a machine to do the work of the computer. ... The machines just described [string machines] do not differ very essentially from computing machines as defined in § 2 [letter machines], and corresponding to any machine of this type a computing machine can be constructed to compute the same sequence, that is to say the

[14] In *A philosophical error in Turing's work*, Gödel points to the next page of *Turing 1936* paper and claims that Turing gives on this second page of section 9 "an argument which is supposed to show that mental procedures cannot go beyond mechanical procedures. However, this argument is inconclusive." (*Collected Works II*, p. 306.) Turing argues there that only finitely many different states of mind affect the mechanical calculation of a human computer; he does *not* make any claim concerning general mental processes as Gödel assumes. (See Webb's *Introductory Note* to *1972*.)

sequence computed by the computer." In the course of his reductive argument, Turing formulates and uses well-motivated constraints.

To articulate these constraints, it is best to use the description of Turing machines as Post production systems. This is appropriate for a number of reasons. Post introduced this description in 1947 to establish that the word-problem of certain Thue-systems is unsolvable. Turing adopted it in 1950 when extending Post's results, but also in 1954 when writing his wonderfully informative and informal essay on solvable and unsolvable problems.[15] This description reflects directly the move in *Turing 1936* to eliminate states of mind for computors[16] in favor of "more physical counterparts" and makes it perfectly clear that Turing is dealing throughout with general symbolic processes. The restricted machine model that results from his analysis obscures that fact.

It is important to recall Turing's central goal, namely, to isolate the most basic steps of computations, that is, steps that need not be further subdivided. This leads to the demand that the configurations, which are directly operated on, be *immediately recognizable* by the computor. This demand and the evident limitation of the computor's sensory apparatus motivate convincingly the central restrictive conditions:

(B) (*Boundedeness*) There is a fixed finite bound on the number of configurations a computor can immediately recognize.

(L) (*Locality*) A computor can change only immediately recognizable (sub-) configurations.[17]

Turing's considerations lead from operations of a computor (in a child's arithmetic book) to operations of a letter machine (on a linear tape). The Turing machine is in the end nothing but, as Gandy put it, a codification of the human

---

[15] Turing expresses the "thesis" as, Every puzzle is equivalent to a substitution puzzle, and remarks on the status of this assertion, "In so far as we know *a priori* what is a puzzle and what is not, the statement is a theorem. In so far as we do not know what puzzles are, the statement is a definition which tells us something about what they are. One can of course define a puzzle by some phrasing beginning, for instance, 'A set of definite rules ...', but this just throws us back on the definition of 'definite rules'. Equally one can reduce it to the definition of 'computable function' or 'systematic procedure'. A definition of any one of these would define the rest." (*Turing 1954*, p. 15)

[16] Turing attributes states of mind to the human computer; machines have "m-configurations".

[17] The conditions are obviously violated in Gödel's equational calculus: the replacement operations quite naturally involve terms of arbitrary complexity.

computer. The analysis secures the generality of mathematical results and, as I emphasized a number of times, their conclusiveness by respecting the broader intellectual context that appealed to effective calculations carried out by humans without invoking higher mental capacities.

V. Concluding remarks.

Turing found his argument mathematically unsatisfactory, as it involved an appeal to *intuition*, i.e., as he put it in *1939*, the ability of "making spontaneous judgments, which are not the result of conscious trains of reasoning". (pp. 208-9) Turing admitted in *1939* that the use of intuition is unavoidable in mathematics.[18] So I assume that we should be wary of "problematic" spontaneous judgment. In Turing's argument, two related problematic steps seem to be involved, namely, i) starting the analysis with calculations in a child's arithmetic book (this is problematic if one wants to consider more general symbolic configurations and procedures) and ii) dismissing, without argument, the two-dimensional character of paper as "no essential of computation". Consequently, the rigorously established result of Turing's analysis can be formulated as follows: Turing machines can carry out the calculations of computors on a linear tape. If the constraints that enter Turing's argument are formulated explicitly as hypotheses, then one can establish: if computors satisfy the boundedness and locality conditions and operate on linear configurations, then string machines codify their computational behavior and Turing's letter machines can carry out their calculations.

If, as Turing allows in *1954*, computors are operating on all kinds of finite discrete configurations, then I locate a deeply problematic appeal to intuition at the precise point where one has to claim that the operations of computors satisfying boundedness and locality conditions can be carried out by "string machines".[19] I consider this claim to be Turing's *central thesis*, as it connects an

---

[18] *Feferman 1988* examines the use of intuition and ingenuity in the context of Turing's work on ordinal logics; see, pp. 129-132.

[19] The term is put in quotation marks, as machines operating on strings can be replaced here by machines operating on more general configurations; cf. my paper with Byrnes, where we consider K-graph machines.

informal concept with a precise mathematical one. The relevant difficulties are rooted in the looseness of the restrictive conditions and the corresponding vagueness of the central thesis. After all, for this general case I have not answered the questions: What are symbolic configurations? What changes can mechanical operations effect?

Without giving a rigorous answer here, some well-motivated ideas can be formulated for computors: (i) they operate on finite configurations; (ii) they recognize immediately a bounded number of different patterns (in these configurations); (iii) they operate locally on these patterns; (iv) they assemble, from the original configuration and the result of the local operation, the next configuration. By exploiting these general ideas, one can attack the problem with a familiar tool, the axiomatic method. It is possible to characterize axiomatically a *Turing Computor* as a special discrete dynamical system and to show that any system satisfying the axioms is computationally equivalent to a Turing machine. (Cf. *Sieg 2002*.) No appeal to a thesis or central thesis is needed; rather, that appeal has been replaced by the standard and non-trivial task of recognizing the correctness of axioms for an intended notion. Here is, I would argue, the proper and unavoidable locus of an appeal to intuition, as it is the case with any mathematical theory that is to capture aspects of our intellectual or physical experience.

Gödel claims in the Gibbs lecture (p. 311) that the state of philosophy "in our days" is to be faulted for not being able to draw in a mathematically rigorous way the philosophical implications of the "mathematical aspect of the situation". I have argued implicitly that not even the "mathematical aspect of the situation" had been clarified in a convincing way. Gödel admits in the 1946 Princeton lecture (p. 152) that certain aspects of the concept of definability "would involve some extramathematical element concerning the psychology of the being who deals with mathematics". That is exactly what Turing recognized and exploited for effective calculability in order to arrive at his analysis of finite procedures carried out by computors.

References

Church, Alonzo

1935 An unsolvable problem of elementary number theory. Preliminary report (abstract); Bull. Amer. Math. Soc. 41, 332-333.

1936 An unsolvable problem of elementary number theory; Amer. J. Math. 58, 345-363; in *Davis 1965*, 89-107

1937 Review of *Turing 1936*; J. Symbolic Logic 2, 42-43.

Dawson, John W.

1997 *Logical Dilemmas – The life and work of Kurt Gödel*; A K Peters, Wellesley.

Davis, Martin

1965 *The Undecidable*; New York

1982 Why Gödel did not have Church's Thesis; Inf. and Control 54, 3-24.

1995 Introductory Note to *Gödel 193?*; in *Collected Works III*, 156-163.

Feferman, Solomon

1988 Turing in the land of O(z); in *Herken 1988*, 113-147.

Gandy, Robin

1980 Church's Thesis and principles for mechanisms; in: Barwise, Keisler, and Kunen (eds.), *The Kleene Symposium*, Amsterdam; 123-148.

1988 The confluence of ideas in 1936; in *Herken 1988*, 55-111.

Gödel, Kurt

1931 Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme; Monatshefte für Mathematik und Physik 38, 173-198, and in *Collected Works I*, 144-195.
Postscriptum (1963); in *Collected Works I*, 195.

1933 The present situation in the foundations of mathematics; lecture delivered at the meeting of the Mathematical Association of America in Cambridge, Massachusetts, December 29-30, 1933, in *Collected Works III*, 45-53.

1934 On undecidable propositions of formal mathematical systems (lecture notes taken by Stephen C. Kleene and J. Barkley Rosser); in *Davis 1965*, 39-71, and *Collected Works I*, 346-369.
Postscriptum (1964), in *Collected Works I*, 369-371.

1936 Über die Länge von Beweisen; Ergebnisse eines mathematischen Kolloquiums 7, 23-24; in *Davis 1965*, 82-3, and *Collected Works I*, 396-8.

193? Undecidable Diophantine propositions; in *Collected Works III*, 164-175.

1939 Notre Dame Lectures on Logic, Spring 1939; cf. Finding Aid in *Collected Works V*, 527-8.

1946 Remarks before the Princeton bicentennial conference on problems in mathematics; in *Davis 1965*, 84-88, and *Collected Works II*, 150-153.

1951 Some basic theorems on the foundations of mathematics and their implications; Gibbs Lecture, *Collected Works III*, 304-323.

1957 Letter to Ernest Nagel of 25 February 1957; in *Collected Works V*, 145-146.
1972 Some remarks on the undecidability results; in *Collected Works II*, 305-306.
1986-2003 *Collected Works, Volumes I – V*; Oxford University Press.

Herbrand, Jacques
1931 On the consistency of arithmetic; in *Logical Writings*; Warren Goldfarb (ed.), Cambridge (Mass.). 1971, 282-298.

Herken, Rolf (ed.)
1988 *The Universal Turing Machine - a half-century survey*; Oxford University Press.

Hilbert, David
1900 Mathematische Probleme; Archiv der Mathematik und Physik 1(1901), 44-63, 213-237.

Hilbert, David & Paul Bernays
1939 *Grundlagen der Mathematik*; vol. II, Springer Verlag, Berlin.

Kleene, Stephen C.
1936 General recursive functions of natural numbers, Mathematische Annalen 112 (5), 727-742; in *Davis 1965*, 237-253.

Odifreddi, Piergiorgio
1990 About logics and logicians – A palimpsest of essays by Georg Kreisel, vol. II: Mathematics. Manuscript

Post, Emil
1947 Recursive unsolvability of a problem of Thue; J. Symbolic Logic 12, 1-11.

Sieg, Wilfried
1994 Mechanical procedures and mathematical experience; in *Mathematics and Mind*, edited by A. George, Oxford University Press, 71-117.
1997 Step by recursive step: Church's analysis of effective calculability, B. Symbolic Logic 3, 154-80.
2002 Calculations by man and machine: conceptual analysis; in *Reflections on the foundations of mathematics*, W. Sieg, R. Sommer, and C. Talcott (eds.); Lecture Notes in Logic 15, Association for Symbolic Logic, 390-409.
2005 Only two letters: the correspondence between Herbrand & Gödel; B. Symbolic Logic 11 (2).

Sieg, Wilfried and John Byrnes
1996 K-graph machines: generalizing Turing's machines and arguments; in *Gödel '96* (P. Hajek, ed.); Lecture Notes in Logic 6, Springer Verlag, 98-119.

Soare, Robert
1999 The history and concept of computability; in *Handbook of Computability Theory*, E.R. Griffor (ed.); Elsevier, 3-36.

Sinaceur, Hourya
2000 Address at the Princeton University bicentennial conference on problems in mathematics, by Alfred Tarski; B. Symbolic Logic 6, 1-44.

Turing, Alan
1936 On computable numbers, with an application to the Entscheidungsproblem; Proc. London Math. Soc. 42, 230-265; in *Davis 1965*, 116-151.
1939 Systems of logic based on ordinals; Proc. London Math. Soc. 45, 161-228; in *Davis 1965*, 155-222.
1948 Intelligent Machinery; written in September 1947, submitted to the National Physical Laboratory in 1948; Machine Intelligence *5*, 3-23.
1950 Computing machinery and intelligence; Mind 59, 433-460.
1950A The word problem in semi-groups with cancellation; Ann. of Math. 52, 491-505.
1954 Solvable and unsolvable problems; Science News 31, 7-23.
1992 *Collected Works of A.M. Turing: Mechanical Intelligence*; D.C. Ince (ed.); North-Holland.

Wang, Hao
1974 *From Mathematics to Philosophy*; London.

Webb, Judson
1990 Introductory Note to Remark 3 of *Gödel 1972*; in *Collected Works II*, 292-304.

Abstract. The identification of an informal concept of "effective calculability" with a rigorous mathematical notion like "recursiveness" or "Turing computability" is still viewed as problematic, and I think rightly so. I analyze three different and conflicting perspectives Gödel articulated in the three decades from 1934 to 1964. The significant shifts in Gödel's position underline the difficulties of the methodological issues surrounding the Church-Turing Thesis.

In his 1964 formulation Gödel sees, quite emphatically, Turing's work as providing a correct analysis of mechanical procedures and thus also of effective calculability. Eight years later he detected, however, "a philosophical error in Turing's work" and attributed a claim to Turing the latter did not maintain when reducing mechanical procedures to machine computations. An analysis of Turing's reduction can serve as the springboard for a different methodological approach, namely, an axiomatic characterization of computability.