

**Oron Shagrir\***

# **Gödel on Turing on Computability**

## **1. Introduction**

In section 9 of his paper, “On Computable Numbers,” Alan Turing outlined an argument for his version of what is now known as the Church–Turing thesis: “the [Turing machine] ‘computable’ numbers include all numbers which would naturally be regarded as computable” [p. 135]. The argument, which relies on an analysis of calculation by an (ideal) human, has attracted much attention in recent years.<sup>1</sup> My aim here is to explain Gödel’s puzzling response to Turing’s argument. On the one hand, Gödel emphasizes at every opportunity that “the correct definition of mechanical computability was established beyond any doubt by Turing” [193?, p. 168]. On the other, in a short note entitled “A philosophical error in Turing’s work,” Gödel [1972a] criticizes Turing’s argument for this definition as resting on the dubious assumption that there is a finite number of states of mind. What are we to make of this apparent inconsistency in Gödel’s response to Turing’s argument? How could Gödel praise Turing’s definition of computability, yet maintain that Turing’s argument for this definition was flawed?

This paper is part of a wider project that aims to establish that, contrary to the widespread assumption, there is no unique concept of that which would naturally be regarded as computing. In earlier papers, I showed that the so-called informal, intuitive, or pre-theoretic notion of computing is understood differently in different contexts. What we take computing to be in the context of computer science<sup>2</sup>

---

\*O. Shagrir, Departments of Philosophy and Cognitive Science, The Hebrew University of Jerusalem.

<sup>1</sup>See, e.g., [Sieg 1994; 2002].

<sup>2</sup>Shagrir [2002].

is different from what we take it to be in the context of physical computation<sup>3</sup> or neuroscience.<sup>4</sup> Further, the pioneers of computability, namely, Church, Gödel, Kleene, Post and Turing, had no shared conception of computing. In the present paper, I will argue that Gödel and Turing anchor finiteness constraints on computability in different considerations. For Turing, these constraints are imposed by limitations on human cognitive capacities, whereas for Gödel, they arise out of foundational work in mathematics in which they are invoked, principally Hilbert’s finitistic program

The paper is structured as follows. I start with a brief review of Turing’s analysis of computability (section 2). I consider the question of why Gödel praises Turing’s analysis (section 3), and then address Gödel’s critique of Turing’s argument (section 4). In the last section (section 5), I reconcile this apparent inconsistency, suggesting that Gödel endorsed the finiteness constraints set down by Turing, but rejected Turing’s claim that they arise out of the limitations imposed by human processing capacities.

## 2. Turing’s Analysis of Computability

Four articles published in 1936 put forward precise mathematical characterizations of, to use Turing’s idiom, that which would naturally be regarded as computable, or what we now call “effective computability.” Church [1936a] characterized the computable in terms of lambda-definability,<sup>5</sup> Kleene [1936] characterized the computable in terms of general recursive functions,<sup>6</sup> Post [1936] characterized it in terms of combinatory processes, and Turing [1936] in terms of Turing machines.<sup>7</sup> Turing wrote the paper when he was a student at Cambridge, and unaware of the other works. He was attempting to solve a problem he had learned of in a course taught by M.H.A. Newman.<sup>8</sup> The problem, referred to by Hilbert and his disciples as the *Entscheidungsproblem*, is now known as the decid-

---

<sup>3</sup>Shagrir and Pitowsky [2003].

<sup>4</sup>Shagrir [forthcoming].

<sup>5</sup>Church started this work in the early 1930’s and completed it with Kleene, who was his student at Princeton; see [Kleene 1981].

<sup>6</sup>This characterization is based on the expansion of primitive recursiveness by Herbrand [1931] and Gödel [1931, 1934]; for some of the history see [Kleene 1981].

<sup>7</sup>The term ‘Turing machine’ first appears in Church [1937a], a review of Turing [1936].

<sup>8</sup>Hodges [1983, pp. 90ff].

ability of first-order predicate logic.<sup>9</sup> To show that the problem has no solution, it is necessary to provide a precise characterization of that which can be solved by an effective (algorithmic) procedure, of which there are infinitely many. Church [1936b] proved the unsolvability of the *Entscheidungsproblem* by utilizing the notions of lambda-definability and recursiveness. Turing came up with a different approach: he reduced the concept of an algorithmic procedure to that of a Turing machine, and then proved that no Turing machine can solve the *Entscheidungsproblem*.<sup>10</sup>

Turing opens his paper with this statement: “The ‘computable’ numbers may be described briefly as the real numbers whose expressions as a decimal are calculable by finite means” [1936, p. 116]. It quickly becomes apparent that Turing associates the intuitive idea of computability with calculability *by humans*.<sup>11</sup> He sets out to provide an explicit definition of computability in terms of (Turing) machine computability, and asserts that “the justification [for this definition] lies in the fact that the human memory is necessarily limited” [p. 117]. Turing then compares “a man in the process of computing a real number” to “a machine which is only capable of a finite number of conditions” [p. 117]; after an informal exposition of the machine’s operations, he states that “these operations include all those which are used in the computation of a number” [p. 118]. The justification for this contention is not produced until section 9. In the interim, Turing provides a mathematical characterization of his machines, proves that the set of these machines is enumerable, shows that there is a universal (Turing) machine, and describes it in detail. He formulates the halting problem, and proves that it cannot be decided by a Turing machine. On the basis of that proof, Turing arrives, in section 11, at his ultimate goal: proving that the *Entscheidungsproblem* is unsolvable.

In section 9 Turing turns to the task of justifying the identification of what would naturally be regarded as computable with Turing machine computability. He adduces three arguments, with the caveat that “all arguments which can be given are bound to

---

<sup>9</sup>Hilbert and Ackermann [1928].

<sup>10</sup>Turing proved the equivalence of Turing machine computability with both lambda-computability and recursiveness in the appendix he added to the 1936 paper after becoming aware of the other work; see [Kleene 1981] for the historical details.

<sup>11</sup>See also [Turing 1947, p. 107].

be, fundamentally, appeals to intuition, and for this reason rather unsatisfactory mathematically” [p. 135]. Turing presents the first argument, on which I will focus in this paper, in part I of section 9, with a modification being added in part III. He characterizes it as a “direct appeal to intuition” and “only an elaboration” of the ideas he had put forward in section 1 [p. 135]. The second argument, in part II, is a proof that all Turing machines can be embedded in first-order logic (“restricted Hilbert functional calculus”). The third argument, in section 10, consists of “examples of large classes of numbers which are computable” [p. 135].

Turing’s argument rests on three ingenious ideas. The first is that computation is defined over symbols and carried out by an agent: this agent is a *human* computer. Thus an adequate characterization of computability should rely on an analysis of human computability. The second idea is that to characterize the computable functions (or numbers, or relations), we should focus on the relevant computational *processes*: “The real question at issue is ‘What are the possible processes which can be carried out in computing a number?’” [p. 135]. The problem, of course, is to characterize the set of infinitely many different computational processes. Here, Turing’s third idea kicks in. Instead of trying to capture all possible processes, Turing offers a small number of constraints on the process of computation. The pertinent constraints must have two properties. First, they must be sufficiently general that their truth will be virtually self-evident. Second, they must be such that when satisfied, the operations of the computing agent can be mimicked by a Turing machine, in other words, the computed function must be Turing-machine computable.

Turing’s argument can be summarized as follows:

**Premise 1 (Thesis H):** A human computer satisfies certain constraints.

**Premise 2 (Turing’s theorem):** The functions computable by a computer satisfying these constraints are Turing-machine computable.

**Conclusion (Turing’s thesis):** The functions computable by a human computer are Turing-machine computable.

What are the constraints on computation? Turing begins his analysis with the observation that “computing is normally done by writing certain symbols on paper. We may suppose this paper is

divided into squares like a child's arithmetic book" [p. 135]. He remarks that the paper is normally two-dimensional, but we can safely assume, with no loss of generalization, that the paper is one-dimensional, i.e., a tape divided into squares. We can also assume that "the number of symbols which may be printed is finite" [p. 135], as we can always use more squares if the string of symbols is lengthy. We can thus assume, with no loss of generality, that we can print at most one symbol in each square.

Turing then sets out a number of more specific constraints. The first is a sort of determinism: "the behavior of the computer at any moment is determined by the symbols which he is observing, and his 'state of mind' at that moment" [p. 136]. He then formulates two restrictions on the conditioning states, i.e., the observed symbols on the tape and the computer's own states of mind, and two on the computer's operations. The first restriction is that "there is a bound  $B$  to the number of symbols or squares which the computer can observe at one moment. If he wishes to observe more, he must use successive observations" [p. 136]. This restriction is motivated by the limitations on our sensory, mainly visual, apparatus, and, in particular, the fact that the visual field contains a limited number of symbols or squares that a human can recognize.

The second restriction is that "the number of states of mind which need be taken into account is finite" [p. 136]. Turing justifies the restriction with the obscure statement that "if we admitted an infinity of states of mind, some of them will be 'arbitrarily close' and will be confused" [p. 136]. He adds that "the restriction is not one which seriously affects computation, since the use of more complicated states of mind can be avoided by writing more symbols on the tape" [p. 136]. In part III of section 9 Turing takes this suggestion a step further, replacing states of mind with a "more physical and definite counterpart" [p. 139], namely, symbolic expressions that can be written on the tape. A crucial assumption is being made here, the assumption that the externalized expressions are finite, but Turing provides little support for it. He seems to appeal to the second argument, found in part II of section 9, according to which any state of a Turing machine is convertible into a formula in first-order predicate logic (and vice versa), but this is just begging the question, for the problematic assumption is the finite character of states of mind, not Turing machines. This will be further addressed below.

Turing proceeds to specify restrictions on the possible operations of a computer, asking us “to imagine the operations the computer performs are to be split up into ‘simple operations’” [p. 136]. “Every such operation,” he asserts, “consists of some change of the physical system consisting of the computer and his tape.” These changes are of three types: changes in the symbols on the tape, changes in the squares being observed, and changes in the computer’s state of mind. As to changes in the symbols, Turing suggests that “the squares whose symbols are changed are always ‘observed’ squares.” As to changes in the squares being observed, Turing suggests that the distance between new and previously-observed squares “does not exceed a certain fixed amount,” i.e., it is bounded. And regarding changes in the computer’s state of mind, he says only that such changes can be conjoined with either of the other two types of changes [p. 137].

Given these restrictions, Turing undertakes to establish the second premise, namely, that we can “construct a [Turing] machine to do the work of this [human] computer” [p. 137]. Turing argues for this only briefly, but the idea is amazingly simple: the constraints imply that there are only finitely many *types* of changes, or atomic operations, that a computer can undergo. It is thus not difficult to agree that these operations can be mimicked by a Turing machine. It is true, as Kleene later explains in his discussion of Turing, that “the human computer is less restricted in behavior than the machine” [1952, p. 377]. In particular, Kleene claims, a human can (a) observe more than one token symbol at a time; (b) perform more complicated atomic acts; (c) use tape that is multi-dimensional; and (d) choose a variety of symbolic representations. But we can reduce these operations to those that can be executed by a Turing machine. Turing takes this reduction to be non-problematic, stating that every action performed by the human computer can be reduced to a finite number of successive steps by a Turing machine. Kleene [1952, pp. 378–381] shows in detail how this reduction is to be carried out with respect to each of the four features of human computing just mentioned. Gandy [1980] and more explicitly, Sieg [2002], formalize the restrictive conditions and provide a detailed mathematical

proof of the second premise.<sup>12</sup> We can thus call this result Turing's theorem.

### 3. Gödel on Turing's Analysis of Computability

It is well known that a statement that seems to be much like the Church–Turing thesis appears in the printed version of Gödel's 1934 Princeton lectures. In the body of the paper, Gödel formulates what is generally taken to be the “easy” part of the Church–Turing thesis: “[primitive] recursive functions have the important property that, for each given set of values of the arguments, the value of the function can be computed by a finite procedure” [p. 348]. In a footnote to this statement, Gödel remarks that “the converse seems to be true if, besides [primitive] recursions [...] recursions of other forms (e.g., with respect to two variables simultaneously) are admitted [i.e., general recursions]. This cannot be proved, since the notion of finite computation is not defined, but it serves as a heuristic principle” [p. 348, note 3]. However, in a letter to Martin Davis (February 15, 1965), Gödel denies that the 1934 paper anticipated the Church–Turing thesis:

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; and in fact the equivalence between my definition and Kleene [1936] is not quite trivial. [Davis 1982, p. 8]

Indeed, Church, who met with Gödel—apparently, early in 1934<sup>13</sup>—commented on the encounter in a letter to Kleene [dated November 29, 1935]:

In regard to Gödel and the notions of recursiveness and effective calculability, the history is the following. In discussion with him [sic] the notion of lambda-definability, it developed that there was no good definition of effective calculability. My

---

<sup>12</sup>Gandy does not invoke Turing's restrictions directly but they can be derived from his conditions on machine computation by adding more restrictions; see [Sieg and Byrnes 1999].

<sup>13</sup>Sieg [1997, p. 160, note 9].

proposal that lambda-definability be taken as a definition of it he regarded as thoroughly unsatisfactory.<sup>14</sup>

But before too long, Gödel's attitude changed. In an unpublished paper, probably from 1938, he writes:

When I first published my paper about undecidable propositions the result could not be pronounced in this generality, because for the notions of mechanical procedure and of formal system no mathematically satisfactory definition had been given at that time. This gap has since been filled by Herbrand, Church and Turing [193?, p. 166].<sup>15</sup>

Thus just four years after having rejected it, Gödel embraces Church's proposal, attributing the "mathematically satisfactory definition" of computability to Herbrand, Church and Turing. Why did Gödel change his mind? It is difficult to say with certainty, but clearly, Turing's work was a significant factor. Initially, he mentions Turing together with Herbrand and Church, but a few pages later Gödel refers to Turing's work as having demonstrated the correctness of the mathematical definition(s): "that this really is the correct definition of mechanical computability was established beyond any doubt by Turing" [p. 168]. More specifically:

[Turing] has shown that the computable functions defined in this way 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  $n_1, \dots, n_r$  on a slip of paper and put the slip 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  $n_1, \dots, n_r$  will be printed on the paper [p. 168].

From this point on, Gödel refers to Turing's work as establishing the "correct definition" of mechanical computability. In his Gibbs Lecture, for instance, he declares:

The greatest improvement was made possible through the precise definition of the concept of finite procedure, which plays a decisive role in these results. There are several different ways

<sup>14</sup>[Davis 1982, p. 9].

<sup>15</sup>Davis dates the article to 1938; see his introduction to [Gödel 193?].

of arriving at such a definition, which, however, all lead to exactly the same concept. The most satisfactory way, in my opinion, 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 [Gödel 1951, pp. 304–305].

And in his 1964 postscript to the 1934 article, Gödel reaffirms:

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.

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’ [pp. 369–370].

Generally speaking, whether we use Turing’s precise definition of computability or other definitions, e.g., general recursiveness, is irrelevant from Gödel’s perspective.<sup>16</sup> After all, they have been *proven* to be extensionally equivalent. Yet it is Turing’s analysis that Gödel consistently appeals to as *accounting for* the correctness of these equivalent definitions of computability. This is a significant point. Turing’s argument has been fully appreciated only recently.<sup>17</sup> Turing’s contemporaries mention the analysis, along other familiar arguments, e.g., the confluence of different notions, quasi-empirical evidence, and Church’s step-by step argument, but do not ascribe

---

<sup>16</sup>In his “Remarks before the Princeton bicentennial conference on problems in mathematics” [Gödel 1946], Gödel, again, does not give priority to Turing computability over the other definitions: “Tarski has stressed in his lecture (and I think justly) the great importance of the concept of general recursiveness (or Turing’s computability)” [p. 150].

<sup>17</sup>Turing’s argument has been praised e.g., Copeland [2002a], Shagrir [2002] and Sieg [2002]; Gandy [1988], Sieg [1994] and Soare [1996] even take it to prove the Church–Turing thesis. The rediscovery of Turing’s analysis is underscored in Martin Davis’s comment [1982, p. 14, note 15] that “this [Turing’s] analysis is still very much worth reading. I regard my having failed to mention this analysis in my introduction to Turing’s paper in Davis [1965] as an embarrassing omission.”

it any special merit.<sup>18</sup> Logic and computer science textbooks from the decades following the pioneering work of the 1930s ignore it altogether.<sup>19</sup> In light of this, Gödel’s praise of Turing’s analysis as the decisive argument for the definition of computability is noteworthy.<sup>20</sup>

Nevertheless, as Gödel does not explain just what it is that makes Turing’s conceptual analysis so convincing, let me begin by clarifying this. First, Gödel favors conceptual analyses over other arguments. He sees the problem of defining computability as “an excellent example [...] of a concept which did not appear sharp to us but has become so as a result of a careful reflection” [Wang 1974, p. 84]. He also says that “Turing’s work gives an *analysis* of the *concept* of ‘mechanical procedure.’ [...] This concept is shown to be equivalent with that of a Turing machine” (my emphasis),<sup>21</sup> and, that it is “absolutely impossible that anybody who understands the question and knows Turing’s definition should decide for a different concept” [Wang 1974, p. 84]. Second, Turing’s analysis is based on an axiomatic approach, the constraints being formulated as basic axioms. This is the approach Gödel recommends in his 1934 conversation with Church, in which he rejects Church’s proposal as “thoroughly unsatisfactory.” Gödel suggests to Church that “it might be possible, in terms of an effective calculability as an undefined notion, to state a set of axioms which would embody the generally accepted

---

<sup>18</sup>Church describes Turing’s identification of effectiveness with Turing-machine computability as “evident immediately” [1937a, p. 43], “an adequate representation of the ordinary notion” [1937b, p. 43], and as having “more immediate intuitive appeal” [1941, p. 41], but does not say it is more convincing than other arguments; see also [Kleene 1952].

<sup>19</sup>Turing’s argument is mentioned in the early days of automata theory, e.g., by McCulloch and Pitts [1943], Shannon and McCarthy [1956], in their introduction to *Automata Studies*, and Minsky [1967, pp. 108–111], who cites it almost in full in his *Finite and Infinite Machines*. Yet even Minsky asserts that the “strongest argument in favor of Turing’s thesis is the fact that [...] satisfactory definitions of ‘effective procedure’ have turned out to be equivalent” [p. 111]. After Minsky, there is no mention of Turing’s argument in logic and computer science textbooks. The two arguments always given for the Church–Turing thesis are the confluence (equivalence) of definitions, and the lack of counterexamples; see, e.g., [Boolos and Jeffrey 1989, p. 20], and [Lewis and Papadimitriou 1981, pp. 223–224].

<sup>20</sup>Gödel doesn’t mention the confluence of definitions and other quasi-empirical evidence as supporting the correctness of the precise definitions in either the Gibbs Lecture or the 1964 postscript to the 1934 paper.

<sup>21</sup>In a footnote to this sentence [note 35, p. 370] Gödel refers readers to Turing’s section 9, where the analysis of computability is carried out. He also mentions “the almost simultaneous paper by E.L. Post [1936].”

properties of this notion, and to do something on that basis” [Davis 1982, p. 9].<sup>22</sup>

Third, Turing’s definition highlights what Gödel takes to be the defining properties of a formal system: being governed by a procedure that is *finite* and *mechanical*. Let me elaborate. There is an important difference in the contexts in which Turing and Gödel invoke the definition of computability. While Turing defines computability in the context of the *Entscheidungsproblem*, Gödel sees the significance of Turing’s definition in the context of the generality of the incompleteness theorems.<sup>23</sup> As he says, the incompleteness results “could not be pronounced in this generality,”<sup>24</sup> until the “gap” was “filled by Herbrand, Church and Turing.” But how exactly has the gap been filled?

When Gödel discusses formal systems, even before 1936, he invokes two essential properties. The property of *being finite* is stressed in 1934, where Gödel opens his address with a characterization of a “formal mathematical system” [p. 346]:

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  $A_1, \dots, A_n$ , and there shall be a finite procedure for determining whether a given formula  $A$  is a meaningful formula or an axiom. [p. 346]

The property of *being mechanical* is spelled out in Gödel’s [1933b] address to the Mathematical Association of America (“The Present

---

<sup>22</sup>Sieg [2002, p. 400] has suggested that Turing’s analysis meets Gödel’s desideratum.

<sup>23</sup>Turing refers to Gödel’s theorems [1936, p. 145], but does not mention the importance of his work for establishing their generality. At the time, Gödel was interested in classes of formulas he took to be decidable (see Goldfarb’s introductory note to Gödel [1932, 1933a]), but was not concerned with the *Entscheidungsproblem*; see also [Gandy 1988, pp. 63–64].

<sup>24</sup>Gödel’s results were published in 1931 under the title “On formally undecidable propositions of *Principia Mathematica* and related systems I.” As the title implies, the results apply to “*Principia Mathematica* and related systems,” and, more precisely, to the formal system  $P$ , which is “essentially the system obtained when the logic of  $PM$  is superposed upon the Peano axioms” [1931, p. 151], and its extensions, which are the “ $\omega$ -consistent systems that result from  $P$  when [primitive] recursively definable classes of axioms are added” [p. 185, note 53].

Situation in the Foundations of Mathematics”). He opens the address with a rough characterization of formal systems, pointing out that the “outstanding feature of the rules of inference being that they are purely formal, i.e., 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” [p. 45]. From Gödel’s perspective, Turing, in defining computability in terms of “a *machine* with a *finite* number of parts” (emphasis added), i.e., in terms of a Turing machine, provides a precise mathematical characterization of a procedure that explicitly satisfies *both* constraints. As such, Turing provides “a precise and unquestionably adequate definition of the general concept of formal system,” hence the incompleteness results are true for “*every* consistent formal system containing a certain amount of finitary number theory.”<sup>25</sup>

It is important to note, however, that although Gödel vacillates between the terms ‘mechanical procedure’ and ‘finite procedure’ when referring to formal systems,<sup>26</sup> he makes clear that they are not synonymous.<sup>27</sup> Moreover, while Gödel takes the finite and mechanical constraints to go hand in hand in the context of a formal system,

---

<sup>25</sup>Or as Gödel puts it in the 1964 postscript to 1934: “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” [pp. 369–370]. See also his Gibbs Lecture: “This requirement for the rules and axioms is equivalent to the requirement that it should be possible to build a finite machine, in the precise sense of a ‘Turing machine,’ which will write down all the consequences of the axioms one after the other” [1951, p. 308].

<sup>26</sup>In 1934 he says “finite procedure” and “finite computation,” in 1938, “mechanical procedure,” in 1951 he reverts to “finite procedure,” and in 1964 to “‘mechanical procedure’ (alias ‘algorithm’ or ‘computation procedure’ or ‘finite combinatorial procedure’).”

<sup>27</sup>When explaining, in the postscript to the 1934 paper, the relevance of Turing’s characterization to the (incompleteness) results, Gödel states that “for any formal system in this sense [of mechanical procedure] there exists one in the sense of page 346 above that has the same provable formulas (and likewise vice versa), provided the term ‘finite procedure’ occurring on page 346 is understood to mean ‘mechanical procedure.’ This meaning, however, is required by the concept of formal system, whose essence it is that reasoning is completely replaced by mechanical operations on formulas” [1964, p. 370]. Further down the same page Gödel reiterates the point, saying that “if ‘finite procedure’ is understood to mean ‘mechanical procedure,’ the question raised in footnote 3 can be answered affirmatively.” Footnote 3 states Gödel’s 1934 assertion that finite computation is included in recursiveness.

in the sense that the procedure must satisfy both constraints, he does not maintain that they coincide in other contexts. In particular, he insists that there are finite procedures that are non-mechanical: “the question of whether there exist finite *non-mechanical* procedures, not equivalent with any algorithm, has nothing whatsoever to do with the adequacy of the definition of ‘formal system’ and of ‘mechanical procedure’” [Gödel 1964, p. 370]. In a footnote he refers to his 1958 article, where he posits procedures that can be construed as fulfilling the finiteness requirement, in that they are constructive, yet are non-mechanical in that they “involve the use of abstract terms on the basis of their meaning” [*ibid.*, note 36].<sup>28</sup> As we will now see, these procedures play an important role in Gödel’s critique of Turing.

#### 4. A Philosophical Error in Turing’s Work

In 1972, Gödel publishes three short remarks on the undecidability results [1972a]. The third is entitled “A philosophical error in Turing’s work.” Gödel declares:

Turing in his [1936, section 9] gives an argument which is supposed to show that mental procedures cannot go beyond mechanical procedures. However, this argument is inconclusive. What Turing disregards completely is the fact that *mind, in its use, is not static, but constantly developing*, i.e., that we understand abstract terms more and more precisely as we go on using them, and that more and more abstract terms enter the sphere of our understanding. There may exist systematic methods of actualizing this development, which could form part of the procedure. Therefore, although at each stage the number and precision of the abstract terms at our disposal may be *finite*, both (and, therefore, also Turing’s number of *distinguishable states of mind*) may *converge toward infinity* in the course of the application of the procedure [p. 306].

What could have motivated Gödel to ascribe this philosophical error to the very analysis he otherwise praises? Fortunately, there is no need to speculate: the answer is explicitly articulated in Gödel’s

---

<sup>28</sup>See also [Gödel 1958], especially [p. 245], and [1972b, pp. 271–275]. These procedures are put forward in the context of expanding Hilbert’s finitary proof methods to include more than just computational procedures; see [Bernays 1935], [Zach 2003].

conversation with Wang [1974, pp. 324–326]. Gödel contends that the incompleteness results imply that “either the human mind surpasses all machines (to be more precise: it can decide more number-theoretical questions than any machine) or else there exist number-theoretical questions undecidable for the human mind” [p. 324].<sup>29</sup> He consistently and unambiguously chooses the former option, averring that “Hilbert was right in rejecting the second alternative” [p. 324], namely, the claim that there are unanswerable mathematical questions. The results, he maintains, indeed indicate that some questions are not decidable in the sense of solvable by means of a formalism, or as we can now put it, by means of a finite machine, viz., a Turing machine. But they do not imply that such questions are “absolutely unsolvable,” namely, cannot be answered by other means. In the 1964 postscript, he expresses this view as follows: “the results mentioned in this postscript do not establish any bounds for the powers of human reason, but rather for the potentialities of pure formalism in mathematics” [p. 370].<sup>30</sup> The human mind, according to Gödel, “*infinitely surpasses the powers of any finite machine*” [1951, p. 310].

This makes it readily apparent what bothers Gödel about Turing’s argument. The concern is that any system constrained by the finiteness conditions set down by Turing cannot transcend the computable. If these constraints indeed apply to the human mind, it cannot surpass the powers of a Turing machine.<sup>31</sup> To avoid this conclusion, Gödel rejects the constraint on the number of states of mind, namely, the finitude of human memory, assuming instead that there are finite (constructive) but non-mechanical mental procedures that enable the mind to infinitely surpass the powers of any finite ma-

---

<sup>29</sup>In his Gibbs Lecture Gödel expresses the dilemma thus: either the “*human mind (even within the realm of pure mathematics) infinitely surpasses the powers of any finite machine, or else there exist absolutely unsolvable diophantine problems of the type specified*” [1951, p. 310].

<sup>30</sup>In a footnote [1972a, p. 306, note 2], Gödel explains that the third remark can be seen as a footnote to this sentence.

<sup>31</sup>In a conversation with Wang, Gödel says: “Turing’s argument becomes valid under two additional assumptions, which today are generally accepted, namely: 1 There is no mind separate from matter. 2. The brain functions basically like a digital computer. (2 may be replaced by: 2’ The physical laws, in their observable consequences, have a finite limit of precision.)” According to Wang, however, “while Gödel thinks that 2 is very likely and 2’ practically certain, he believes that 1 is a prejudice of our time, which will be disproved scientifically” [Wang 1974, p. 326]. Thus Gödel apparently holds that Turing’s constraints, or a version of them, apply to the brain, but not the mind.

chine. Gödel admits that “construction of a well-defined procedure which could actually be carried out (and would yield a non-recursive number-theoretic function) would require a substantial advance in our understanding of the basic concepts of mathematics” [1972a, p. 306]. He mentions two possible examples of such procedures: “the process of forming stronger and stronger axioms of infinity in set theory,” and “the process of systematically constructing, by their distinguished sequences  $\alpha_n \rightarrow \alpha$ , all recursive ordinals  $\alpha$  of the second number-class” [1972a, p. 306].<sup>32</sup>

Whether Turing, in 1936, sought to provide an argument to the effect that mental procedures cannot go beyond mechanical procedures, is open to question, as is the degree to which his analysis of computability motivated this claim, if he indeed made it.<sup>33</sup> Also open to question is whether we can avoid this mechanistic conclusion by invoking non-mechanical procedures.<sup>34</sup> Our question here, however, is different, and pertains to the duality in Gödel’s response to Turing. On the one hand, as we saw, Gödel repeatedly praises Turing’s analysis of computability, saying it produced a “correct and unique” definition of “the concept of mechanical” in terms of “the sharp concept of ‘performable by a Turing machine’” [Wang 1974, p. 84]. On the other, he deems “fallacious” Turing’s “alleged proof” for the “equivalence of minds and machines,” which rests on the very same analysis [Wang 1974, p. 325]. How could Gödel praise Turing’s

---

<sup>32</sup>For more examples, see [Wang 1974, pp. 325–326].

<sup>33</sup>Although it has been claimed that Turing [1950] endorsed a mechanistic view of the mind, namely, the view that thought consists of finite and mechanical procedures, it is fairly clear that this claim is incorrect. It is also doubtful that Gödel attributes this view to Turing. What Gödel apparently attributes to Turing is the weaker view that “mental procedures cannot go beyond mechanical procedures” in the more behavioristic sense that the behavior generated by mental procedures would not be distinguishable from that generated by mechanical procedures. Gödel might have thought that this weaker view, which Turing apparently advances in his 1950 *Mind* article, was expressed in the 1936 article, and motivated by the claim (which Gödel does attribute to Turing) that the finiteness constraints on mechanical procedures are general limitations on the human mind. Whether Turing really made the latter claim is a matter of controversy, and is briefly discussed in the next section [see note 40].

<sup>34</sup>Webb [1980] argues that even if the mind is capable of infinitely many states “it would still have to be shown that we could make effective use of all these states” [p. 223]. Kleene [1987] argues that since the methods for converging to infinity are not explicitly stated, what Gödel is contemplating is just “pie in the sky” [p. 494].

definition but not Turing's argument? How could he invoke Turing's analysis of computability and reduction of "the concept of finite procedure to that of a machine with a finite number of parts," while at the same time rejecting what seems to be a key element of this analysis? In his introductory note to Gödel [1972a], Webb suggests that Gödel was of the opinion that "all Turing was really *analyzing* was the concept of 'mechanical procedure,' but in his *arguments* for the adequacy of his analysis he overstepped himself by dragging in the mental life of a human computer" [1990, p. 302]. This, I think, is exactly right, but the question, as Webb notes, remains: how could Gödel "enjoy the generality conferred on his results by Turing's work, despite the error of its ways"? [1990, p. 293].

## 5. Making (More) Sense of Gödel's Comments

An obvious way to reconcile the disparity is to appeal to the distinction between Turing's argument in part I of section 9 (henceforth, type I argument), where Turing assumes that the "the number of states of minds which need be taken into account is finite" [p. 136], and its modification in part III of section 9 (henceforth, type III), in which "we avoid introducing the 'state of mind' by considering a more physical and definite counterpart of it" [p. 139], i.e., written symbolic instructions. Thus, according to Webb, Feferman maintains that "Gödel rejected only Turing's type I argument, while accepting his 'more physical' type III argument" [1990, p. 297], and Webb concurs that "Gödel took issue with [...] [Turing's] type I argument" [1990, p. 302]. If this suggestion is correct, Gödel rejects the type I argument because it anchors the finite and mechanical nature of computational procedures in the assumption that human memory is necessarily limited, and, in particular, the number of states of mind is *bounded*. He embraces the type III argument because it does not rest on this assumption, but on a "more physical and definite counterpart of it."

What must still be explained, however, is how the "more physical and definite counterpart" anchors the finite and mechanical nature of the computation procedure. Is it not possible that there is a computation procedure that is either non-finite or non-mechanical? As we saw, Gödel himself alludes to constructive procedures, which can be written down in finitely many symbols, but whose execution is non-mechanical inasmuch as it appeals to the symbols' meanings. There

can also be a mechanical procedure, consisting of infinitely many conditions, that produces a solution for the halting problem.<sup>35</sup> So what could justify the finite and mechanical nature of the procedure, if not the fact that human memory is necessarily limited?

One answer might be that the finite and mechanical nature of the computation procedure is entailed by Turing's other constraints. Sieg [2002], taking this tack, has reformulated Turing's constraints, dropping the requirement about states of mind and memory altogether.<sup>36</sup>

The claim being made, then, is that Turing's type III argument simply highlights the fact that this requirement is redundant. But this answer is unsatisfactory. First, it has been shown that the other constraints do not alone suffice to establish the identification of effectiveness and Turing-machine computability.<sup>37</sup> Second, this answer does not explain Gödel's response to Turing. For if the requirement on states of mind is redundant, then, from Gödel's perspective, the other constraints should suffice to establish that the mind cannot surpass the powers of a Turing machine.

A second answer might be that although Gödel denies that human memory is necessarily limited, he agrees that the number of states of mind that must be taken into account, *when calculating*, is finite. The latter assumption is quite weak, and suffices, with the other constraints, to establish the identity of mechanical computability and Turing-machine computability.<sup>38</sup> Here, the claim being made is that the type III argument highlights the fact that the weaker assumption suffices, and there is no need for the stronger-and

---

<sup>35</sup>See, e.g., [Shagrir and Pitowsky 2003]. A simpler example is a machine with infinitely many states, where each state  $n$  encodes the self-halting state of the  $n^{\text{th}}$  Turing machine. Thus given input  $n$ , the machine moves  $n$  states (in  $n$  steps) and produces the self-halting state of the  $n^{\text{th}}$  Turing machine.

<sup>36</sup>In his recent presentation of Turing's analysis, Sieg [2002, p. 396] invokes only two restrictive conditions: boundedness ("there is a fixed bound on the number of configurations a computer [human computer] can immediately recognize"), and locality ("a computer can change only immediately recognizable (sub-) configurations").

<sup>37</sup>See [Shagrir and Pitowsky 2003], who demonstrate that without the two general constraints, finite-and-mechanical procedure, and finitely many steps in a finite time, Turing-machine computability can be surpassed. An infinite (yet mechanical!) procedure that encodes the infinitely many self-halting states of all machines can be used to compute the self-halting problem without violating boundedness and locality [see too note 35].

<sup>38</sup>This was suggested to me by Jonathan Yaari.

false-assumption that limits the number of states of mind in general. There are two problems with this proposal. One is that Turing himself suggests the weaker assumption, saying that “the number of states of mind *which need be taken into account* is finite” [p. 136, my emphasis].<sup>39</sup> So it is unclear what motivates Gödel to emphasize a stronger reading of Turing.<sup>40</sup> The second pertains to the source of the finitude: why is the number of states of mind that must be taken into account bounded? If the number of states of mind in general may be infinite, or at least unbounded, is there any reason to think that with respect to calculation the number of states of mind is bounded? It would seem that an implicit assumption is being made here, to the effect that that the process of (human) calculation is associated with a cognitive ‘module’ with a finite number of states of mind. But this assumption is exceedingly contentious, and renders Turing’s analysis, and Gödel’s reasons for extolling it, vulnerable to obvious objections.

Another proposal is that the finite and mechanical nature of the procedure lies in some publicity constraint. Thus Kleene [1987] argues that whether or not the mental states converge to infinity has “no bearing on what number-theoretic functions are effectively calculable” [p. 493]. The idea of ‘effective calculability’ or an ‘algorithm’ involves a set of instructions that is fixed in advance. This condition is motivated by a publicity constraint, namely, that it must be possible “to convey a complete description of the effective procedure or algorithm by a finite communication, in advance of performing computations in accordance with it. My version of the Church–Turing thesis is thus the ‘*Public-Processes Version*’” [pp. 493–494]. The public character of the procedure is also emphasized by Sieg [2006], who contends that “it was the normative demand of radical intersub-

---

<sup>39</sup>This point is emphasized in [Shagrir and Pitowsky 2003]; see also [Sieg 2006], who writes: “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” [note 12].

<sup>40</sup>As we saw, the stronger assumption comes out in Turing’s statement in section 1 that the justification for the identification of effective computability with Turing-machine computability “lies in the fact that the human memory is necessarily limited.” Moreover, Turing clearly does not intend to provide a very different argument in section 9, as he says there that his analysis “is only an elaboration of the ideas of [section 1]” [p. 135]. Gödel’s reading may have been influenced by Turing’s 1950 *Mind* article [see note 33].

jectivity between humans that motivated the steps from axiomatic to formal systems.”

It may well be that Gödel accepts the type III argument because it does not rest on dubious assumptions about the human mind, but on the “normative demand of radical intersubjectivity.” The question, however, is whether this public accessibility constraint must be explicated in terms of finiteness, i.e., “finite communication.” Why is it not possible for the relevant procedures to be conveyed through “infinite” communication? Sieg [2006] argues that the answer lies in the “limitations of [...] [human] processing capacities, when proceeding mechanically,” which is precisely the reason “Turing most appropriately brings in human computers in a crucial way.” Sieg thus concludes that “in a deep sense neither Church nor Gödel recognized the genuinely distinctive character of Turing’s analysis,” i.e., that the calculations are “carried out programmatically by human beings.” If this is right, then, as Webb puts it, there is no essential difference between type I and type III arguments: “Turing has *one* basic argument, which is presented in Section 1 [...] and whose central premise is ‘the fact that the *human* memory is necessarily limited’” [1990, p. 302]. Webb thus concludes that on Gödel’s premises, even Turing’s type III argument is unacceptable. Gödel accepts it, Webb argues, only because he mistakenly interprets it to refer to mechanical operations shorn of human aspect; or as Webb puts it, “in reflecting on this argument [type III] in 1972, Gödel forgot that the word ‘computer’ here meant only what that word meant in 1936: a person doing calculations” [1990, p. 302].<sup>41</sup>

On the picture described by these critics, then, Gödel’s reading of Turing is confused. Gödel, they claim, rejects Turing’s type I argument, yet embraces Turing’s type III argument, though the two arguments are essentially the same. He rejects the type I argument because he takes it to rest on the questionable assumption that the number of states of mind is bounded, yet ignores Turing’s assertion that this assumption is made solely with respect to the context of mechanical calculation. And Gödel embraces the type III argument even though it too is premised on the very limitations on human processing he rejects.

I think we can be more charitable to Gödel. My suggestion is that we take him as holding the view that the finite and mechanical

---

<sup>41</sup>See also [Hodges 1983, p. 105].

character of computation is not matter of the human condition, but of the epistemic role of computation in the foundations of mathematics, and, in particular, in the finitistic program of Hilbert. Let me explain. There is a major difference between the historical contexts in which Turing and Gödel worked. Turing tackled the *Entscheidungsproblem* as an interesting mathematical problem worth solving; he was hardly aware of the fierce foundational debates.<sup>42</sup> Gödel, on the other hand, was passionately interested in the foundations of mathematics. Though not a student of Hilbert, his work was nonetheless deeply entrenched in the framework of Hilbert's finitistic program, whose main goal was to provide a meta-theoretic finitary proof of the consistency of a formal system "containing a certain amount of finitary number theory."<sup>43</sup> In this foundational context, a formal mathematical system is a system governed by a procedure that is finite and mechanical.<sup>44</sup> A computation procedure is just another name for this finite and mechanical procedure. Thus the procedure's finite and mechanical nature is a given and not open to question. Its finite and mechanical nature is underwritten by its role in the foundational project, which is defining a formal mathematical system.<sup>45</sup>

Turing's error, on this account, is anchoring the procedure's finite and mechanical character in the human condition, specifically, in the number of states of mind. Turing's analysis, according to Gödel, does not establish that a computation procedure is a finite and mechanical procedure, for this is not questionable at all. As we saw, Gödel

---

<sup>42</sup>The *Entscheidungsproblem*, while described by Hilbert and Ackermann [1928] as the most fundamental question in mathematical logic, is peripheral to Hilbert's program.

<sup>43</sup>See [Bernays 1935] and [Zach 2003]. Already in the incompleteness article, Gödel writes that the second result does not "contradict Hilbert's formalistic viewpoint. For this viewpoint presupposes only the existence of a consistency proof in which nothing but finitary means of proof is used, and it is conceivable that there exist finitary proofs that *cannot* be expressed in the formalism  $P$  (or of  $M$  or  $A$ )" [1931, p. 195]. See also [Gödel 1933c, 1958 and 1972b], where Gödel produces a consistency proof by means of finite but non-mechanical procedures.

<sup>44</sup>See [Sieg 1994] for an historical survey, going back to Leibniz, of the relations between computation and formal systems. Gödel [1933b, pp. 50–52] discusses the constraints Hilbert's program imposes on the definition of a formal system.

<sup>45</sup>Thus in a footnote added to his 1946 remarks for the Davis anthology, Gödel defines a computable function  $f$  in a formal system  $\mathcal{S}$  "if there is in  $\mathcal{S}$  a computable term representing  $f$ " [p. 84]. A similar definition is advanced in [Gödel 1936], [Church 1936a], and [Hilbert and Bernays 1939].

used the notion of a *mechanical* and *finite* procedure/computation before he encountered Turing's analysis, at about the time he rejects Church's proposal as "thoroughly unsatisfactory." By Gödel's lights, even if it turns out that a human has infinite memory or can carry out infinitely many steps in finite time, this would not change either the definition of computability or that of a formal system. Indeed, Gödel thinks that human memory is not limited, and that human thought is more powerful than any finite machine, but this implies nothing about the finite and mechanical character of the computational procedures and formal systems.<sup>46</sup>

I am not suggesting that Gödel sees no connection between mechanical computability and human computability. Quite the contrary: the epistemic context requires that a *human* be able, at least in principle, to follow the computation procedure, i.e., to check whether a configuration of symbols constitutes a formal proof or not. Gödel praises Turing precisely for this, for analyzing the concept of a human who follows a finite and mechanical procedure. This analysis indeed invokes a set of constraints that are specific to a human calculator, and are the basis for the definition of Turing-machine computability. The analysis is correct because to establish that definition, it suffices to assume that the human follows a finite and mechanical procedure; i.e., that he or she follows a finite set of instructions, execution of the instructions requires no reference to the meanings of the symbols, and the process itself, if it terminates at all, consists of finitely many steps. Turing's error, according to Gödel, is to assume, in addition, that the finite and mechanical nature of the procedure lies in the human condition, i.e., in limitations on human processing capacities.<sup>47</sup>

On the picture I advance here, the finite and mechanical character of computation is rooted in its role in defining a formal system. But it could be argued that even granting my construal of the characterization of computation as finite and mechanical, no account of

---

<sup>46</sup>Gödel mentions the possibility of accelerated processing in [Wang 1974, p. 325]. See also [Copeland 2002b] and [Shagrir and Pitowsky 2003], who point out that acceleration increases computational power, without violating the Church–Turing thesis.

<sup>47</sup>See [Gandy 1980] and [Sieg 2002], who do away with boundedness in the context of physical machines. See also [Shagrir 2002] for (non-physical) machines that violate locality. However, even here the more general constraints must be preserved; otherwise, non-Turing-machine computability emerges [Shagrir and Pitowsky 2003].

the finite character of the epistemic constraints has been provided. We are still in the dark as to why the procedure governing a formal system must be finite and mechanical. This calls to mind Sieg's notion of the 'stumbling block' in the analysis of computation. On Sieg's account, Turing's major achievement is bootstrapping from the circularity of the appeal to the finite nature of computation.<sup>48</sup> Turing justifies both the restrictive conditions on the computation procedures and the epistemic conditions on formal systems by anchoring the finiteness conditions in human processing capacities. It is precisely at this point, and for this reason, Sieg claims, that "Turing most appropriately brings in human computers in a crucial way and exploits the limitations of their processing capacities, when proceeding mechanically" [Sieg 2006]. Failing to appreciate this reasoning, Church and Gödel failed to recognize the "genuinely distinctive character" of Turing's analysis.

I agree with Sieg that Turing's analysis provides an elegant account of the finite character of the relevant epistemic constraints. I also agree that Gödel does not provide an alternative account. But I do not see the necessity for such an account. Hilbert's program is *one* attempt to secure mathematics; it is not the only way to secure mathematics. Had it succeeded, mathematics would have been "secured." Because, as it turned out, the program failed, there is still room, at least in principle, for another such project.<sup>49</sup> If my explication of Gödel on Turing is on track, then being finite and mechanical is not a condition on *any* epistemic procedure? We invoke the notion of finite-and-mechanical procedures because we think it has epistemic value. Hence, if a question of justification arises at all, it is not that which Sieg poses, namely, what justifies the finite and mechanical nature of the procedure. The relevant question is whether the finiteness conditions have epistemic value at all: whether what is done by means of finite and mechanical procedures is thereby to be considered "secured." This has been debated,<sup>50</sup> but Turing's answer

---

<sup>48</sup>Sieg [1994; 1997] presents this circularity in the context of Church's step-by-step argument. Church defines computability in terms of representability in a formal system, but then assumes that the basic operations of the system must be finite, e.g., recursive.

<sup>49</sup>In fact, Gödel's 1958 and 1972b papers can be read as suggesting a renewed and expanded Hilbertian program; see also [Bernays 1935].

<sup>50</sup>Thus strict finitists would object that computation procedures can be applied to any numeral whatsoever. It has also been claimed, more recently, that com-

is no more edifying than Gödel's. Gödel's stance that the human mind infinitely surpasses any finite machine is consistent with the idea that the finiteness constraints do guarantee knowledge.

In closing, let me reiterate that my aim here was not to defend Gödel's interpretation of Turing. Turing's comments are ambiguous, and subject to conflicting interpretations. It may even be the case that his position is not all that different from the position I ascribe to Gödel. My aim was to make sense of Gödel's seemingly conflicted response on Turing's analysis. Gödel praised Turing for his analysis of an ideal human who calculates by means of finite and mechanical procedures. He was critical of what he deemed Turing's superfluous assumption that the finite and mechanical character of computation is somehow anchored in limitations on human cognitive capacities.<sup>51</sup>

## References

- Bernays, P. [1935] "Hilbert's Investigations of the Foundations of Arithmetic", the German text is from Hilbert's *Gesammelte Abhandlungen*, vol. 3, pp. 196–216, translated and reprinted in the *Bernays Project*, <<http://www.phil.cmu.edu/projects/bernays/>>.
- Boolos, G.S. and Jeffrey, R.C. [1989], *Computability and Logic*, 3<sup>rd</sup> edition, Cambridge: Cambridge University Press.
- Church, A. [1936a] "An Unsolvable Problem of Elementary Number Theory", *American Journal of Mathematics* **58**, 345–363; reprinted in [Davis 1965, pp. 88–107].
- Church, A. [1936b] "A Note on the *Entscheidungsproblem*", *Journal of Symbolic Logic* **1**, 40–41; reprinted in [Davis 1965, pp. 108–115].
- Church, A. [1937a], Review of Turing [1936], *Journal of Symbolic Logic* **2**, 42–43.
- Church, A. [1937b], Review of Post [1936], *Journal of Symbolic Logic*, **2**, 43.
- Church, A. [1941], *The Calculi of Lambda-Conversion*, Princeton: Princeton University Press.

---

putation procedures are of value to proof theory only if they are of polynomial complexity, "reasonable" length, and so on.

<sup>51</sup>This research was supported by The Israel Science Foundation, grant 857/03-07.

- Copeland, J.B. [2002a], “The Church–Turing Thesis”, in *The Stanford Encyclopedia of Philosophy*, (E. Zalta ed.), <<http://plato.stanford.edu>>.
- Copeland, J.B. [2002b], “Accelerating Turing Machines”, *Minds and Machines* **12**, 281–301.
- Davis, M. (ed.) [1965], *The Undecidable: Basic Papers on Undecidable Propositions, Unsolvable Problems and Computable Functions*, New York: Raven.
- Davis, M. [1982], “Why Gödel didn’t have Church’s Thesis”, *Information and Control* **54**, 3–24.
- Gandy, R. [1980], “Church’s Thesis and Principles of Mechanisms”, in *The Kleene Symposium*, (J. Barwise, H.J. Keisler, and K. Kunen eds.), Amsterdam: North–Holland, pp. 123–148.
- Gandy, R. [1988], “The Confluence of Ideas in 1936”, in [Herken 1988, pp. 51–102].
- Gödel, K. [1931], “On Formally Undecidable Propositions of Principia Mathematica and Related Systems”, *Monatshefte für Mathematik und Physik* **38**, 173–198; in *Collected Works I*, pp. 144–195.
- Gödel, K. [1932], “A Special Case of the Decision Problem for Theoretical Logic”, in *Collected Works*, I, pp. 231–235.
- Gödel, K. [1933a], “On the Decision Problem for the Functional Calculus of Logic”, in *Collected Works*, I, pp. 307–327.
- Gödel, K. [1933b], “The Present Situation in the Foundations of Mathematics”, in *Collected Works*, III, pp. 45–53.
- Gödel, K. [1933c], “On Intuitionistic Arithmetic and Number Theory”, in *Collected Works*, I, pp. 287–295.
- Gödel, K. [1934], “On Undecidable Propositions of Formal Mathematical Systems”, in *Collected Works*, I, pp. 346–369.
- Gödel, K. [193?], “Undecidable Diophantine Propositions”, in *Collected Works*, III, pp. 164–175.
- Gödel, K. [1946] “Remarks before the Princeton Bicentennial Conference on Problems in Mathematics”, in [Davis 1965, pp. 84–88], and in *Collected Works*, II, pp. 150–153.
- Gödel, K. [1951], “Some Basic Theorems on the Foundations of Mathematics and their Implications”, in *Collected Works*, III, pp. 304–323.

- Gödel, K. [1958], “On a Hitherto Unutilized Extension of the Finitary Standpoint”, in *Collected Works*, II, pp. 241–251.
- Gödel, K. [1964], “Postscriptum to Gödel 1934”, in *Collected Works*, I, pp. 369–371.
- Gödel, K. [1972a], “Some Remarks on the Undecidability Results”, in *Collected Works*, II, pp. 305–306.
- Gödel, K. [1972b], “On an Extension of Finitary Mathematics which has not yet been Used”, in *Collected Works*, II, pp. 271–280.
- Gödel, K. [1986–1995], *Collected Works*, vol. I–III, (S. Feferman, et al. eds.), Oxford: Oxford University Press.
- Goldfarb, W.D. [1986], “Introductory Note to Gödel 1932 and 1933[a]”, in *Collected Works*, I, pp. 226–231.
- Herbrand, J. [1931], “On the Consistency of Arithmetic”, in *Jacques Herbrand Logical Writings*, (W.D. Goldfarb ed.), Cambridge MA: Harvard University Press [1971], pp. 282–298.
- Herken, R. (ed.) [1988], *The Universal Turing Machine: A Half-Century Survey*, Oxford: Oxford University Press.
- Hilbert, D. and Ackermann, W. [1928], *Grundzüge der Theoretischen Logik*, Berlin: Springer-Verlag.
- Hilbert, D. and Bernays, P. [1939], *Grundlagen der Mathematik II*, Berlin: Springer-Verlag.
- Hodges, A. [1983], *Alan Turing: The Enigma*, New York: Simon and Schuster.
- Kleene, S.C. [1936], “General Recursive Functions of Natural Numbers”, *Mathematische Annalen* **112**, 727–742; reprinted in [Davis 1965, pp. 236–253].
- Kleene, S.C. [1952], *Introduction to Metamathematics*, Amsterdam: North-Holland.
- Kleene, S.C. [1981], “Origins of Recursive Function Theory”, *Annals of the History of Computing* **3**, 52–67.
- Kleene, S.C. [1987], “Reflections on Church’s Thesis”, *Notre Dame Journal of Formal Logic* **28**, 490–498.
- Lewis, H.R. and Papadimitriou, C.H. [1981], *Elements of the Theory of Computation*, Eaglewood Cliffs, NJ: Prentice-Hall.

- McCulloch, W.S. and Pitts, W. [1943], “A Logical Calculus of the Ideas Immanent in Nervous Activity”, *Bulletin of Mathematical Biophysics* **5**, 115–133.
- Minsky, M.L. [1967], *Computation: Finite and Infinite Machines*, Eaglewood Cliffs NJ: Prentice–Hall.
- Post, E.L. [1936], “Finite Combinatory Processes—Formulation I”, *Journal of Symbolic Logic* **1**, 103–105; reprinted in [Davis 1965, pp. 288–291].
- Shagrir, O. [2002], “Effective Computation by Humans and Machines”, *Minds and Machines* **12**, 221–240.
- Shagrir, O. and Pitowsky, I. [2003], “Physical Hypercomputation and the Church–Turing Thesis”, *Minds and Machines* **13**, 87–101.
- Shannon, C.E. and McCarthy, J. (eds.) [1956], *Automata Studies*, Annals of Mathematics Studies 34, Princeton: Princeton University Press.
- Sieg, W. [1994], “Mechanical Procedures and Mathematical Experience”, in *Mathematics and Mind*, (A. George ed.), Oxford: Oxford University Press, pp. 71–117.
- Sieg, W. [1997], “Step by Recursive Step: Church’s Analysis of Effective Calculability”, *Bulletin of Symbolic Logic* **2**, 154–180.
- Sieg, W. [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, Natick, MA: Association for Symbolic Logic, pp. 390–409.
- Sieg, W. [2006], “Gödel on Computability”, *Philosophia Mathematica*, (forthcoming).
- Sieg, W. and Byrnes, J. [1999], “An Abstract Model for Parallel Computations: Gandy’s Thesis”, *Monist* **82**, 150–164.
- Soare, R.I. [1996], “Computability and Recursion”, *Bulletin of Symbolic Logic* **2**, 284–321.
- Turing, A.M. [1936], “On Computable Numbers, with an Application to the *Entscheidungsproblem*”, *Proceedings of the London Mathematical Society* **42**, 230–265; correction in **43**(1937), 544–546; reprinted in [Davis 1965, pp. 115–154]; page numbers refer to the [1965] edition.

- 
- Turing, A.M. [1947], “Lecture to the London Mathematical Society on 20 February 1947”, in *A.M. Turing’s ACE Report of 1946 and Other Papers*, (B.E. Carpenter and R.W. Doran eds.), Cambridge MA: MIT Press. [1986], pp. 106–124.
- Turing, A.M. [1950], “Computing Machinery and Intelligence”, *Mind* **59**, 433–460.
- Wang, H. [1974], *From Mathematics to Philosophy*, London: Routledge & Kegan Paul.
- Webb, J.C. [1980], “Mechanism, Mentalism and Metamathematics”, Dordrecht: D. Reidel.
- Webb, J.C. [1990], “Introductory Note to Remark 3 of Gödel 1972[a]”, in *Collected Works*, II, pp. 292–304.
- Zach, R. [2003], “Hilbert’s Program”, in *The Stanford Encyclopedia of Philosophy*, (E. Zalta ed.), <<http://plato.stanford.edu/>>.

# Index

- Ackermann, W., 3, 20
- Bernays, P., 13, 20, 22
- Boolos, G.S., 10
- Byrnes, J., 7
- Church, A., 2, 3, 7, 8, 10, 11, 19–22
- Copeland, B.J., 9, 21
- Davis, M.D., 7–9, 11, 20
- Gandy, R.O., 6, 7, 9, 11, 21
- Gödel, K., 1, 2, 7–23
- Goldfarb, W., 11
- Herbrand, J., 2, 8, 11
- Hilbert, D., 2, 3, 14, 20
- Hodges, A., 2, 19
- Jeffrey, R.C., 10
- Kleene, S.C., 2, 3, 6, 7, 10, 15, 18
- Leibniz, G., 20
- Lewis, H.R., 10
- McCarthy, J., 10
- McCulloch, W.S., 10
- Minsky, M.L., 10
- Newman, M.H.A., 2
- Papadimitriou, C.H., 10
- Pitowsky, I., 2, 17, 18, 21
- Pitts, W., 10
- Post, E.L., 2, 10
- Shagrir, O., 1, 2, 9, 17, 18, 21
- Shannon, C.E., 10
- Sieg, W., 1, 6, 7, 9, 11, 17–22
- Soare, R.I., 9
- Tarski, A., 9
- Turing, A.M., 1–6, 8–23
- Wang, H., 10, 14, 15, 21
- Webb, J.C., 15, 16, 19
- Zach, R., 13, 20