"The Church-Turing "Thesis" as a Special Corollary of Gödel's Completeness Theorem," in *Computability: Turing, Gödel, Church, and Beyond*, B. J. Copeland, C. Posy, and O. Shagrir (eds.), MIT Press (Cambridge), 2013, pp. 77-104.

Saul A. Kripke

4 The Church-Turing "Thesis" as a Special Corollary of Gödel's Completeness Theorem[1]

Saul A. Kripke

Traditionally, many writers, following Kleene (1952), thought of the Church-Turing thesis as unprovable by its nature but having various strong arguments in its favor, including Turing's analysis of human computation. More recently, the beauty, power, and obvious fundamental importance of this analysis—what Turing (1936) calls "argument I"— has led some writers to give an almost exclusive emphasis on this argument as *the* unique justification for the Church-Turing thesis. In this chapter I advocate an alternative justification, essentially presupposed by Turing himself in what he calls "argument II." The idea is that computation is a special form of mathematical deduction. Assuming the steps of the deduction can be stated in a first-order language, the Church-Turing thesis follows as a special case of Gödel's completeness theorem (first-order algorithm theorem). I propose this idea as an alternative foundation for the Church-Turing thesis, both for human and machine computation. Clearly the relevant assumptions are justified for computations presently known. Other issues, such as the significance of Gödel's 1931 Theorem IX for the *Entscheidungsproblem*, are discussed along the way.

4.1 The Previously Received View and More Recent Challenges

It was long a commonplace in most writings on computability theory that the Church-Turing thesis, identifying the several mathematical definitions of recursiveness or computability with intuitive computability, was something that could be given very considerable intuitive evidence, but was not a precise enough issue to be itself susceptible to mathematical treatment (see, for example, Kleene's classic treatise [1952, section 62, 317–323]).[2] Some reservations to this view are already expressed, albeit relatively weakly, in Shoenfield (1967), section 6.5 (see 119–120), though the rest of the section follows Kleene (I owe this observation to Stewart Shapiro). Later in 1993, Shoenfield expressed himself more explicitly (see Shoenfield 1993, 26). What we would need, according to him, would be some self-evident axioms for, or

a characterization of, intuitive computability, implying that all intuitively computable functions were Church-Turing recursive (computable). Harvey Friedman, in conversation with me (confirmed by Shapiro), had always taken a similar view.[3]

I too had always felt that the issue of Church's thesis wasn't one that was obviously not a mathematical one, susceptible of proof or disproof, but could be a genuine technical problem, though I thought it may be very difficult.[4] I do think the axiomatic approach may well be viable, and indeed have said so; but I'm going to give a different approach to the issue of Church's thesis here.

Two things had always led me to doubt what I have called the "conventional view" (but see note 3, this paper), as quoted from Kleene. First, one half of the Church-Turing thesis, that the recursive or Turing-computable functions are all in fact effectively calculable, can I think easily be established simply on the basis of an intuitive notion of calculability. No axiomatization is required, just a direct argument. And the argument should be regarded as a rigorous intuitive *proof* of its result.[5]

Second, it always seemed to me, as against those who held that there isn't any actual mathematical question here but just an explication, that there is after all another thesis that might have seemed to have a lot of evidence for it in the old days. That is, that the effectively calculable functions simply should be identified with the primitive recursive functions.[6] Ackermann (1928) actually *disproved* this thesis, by exhibiting a function that was obviously effectively calculable, but not primitive recursive. Now, clearly the Church-Turing thesis could in principle be disproved in the same way.[7,8] Given that this is so, how can we say that a mathematical question is not involved, since there can be a disproof?

I resumed thinking about the subject of mathematical arguments for the Church-Turing thesis as a reaction to Soare (1996). In his admirable survey of the history and philosophy of the subject, he proposed that the subject traditionally called "recursion theory" should be renamed "computability theory," and that correspondingly such terms as "recursively enumerable" should be renamed "computably enumerable." This reform has been generally adopted by writers on the subject.[9] However, I am probably too old and set in my ways, and do not propose here to stick to the reform. Also, in agreement with Slaman, a leading figure in the subject whom I gather does not propose to adopt the reform, I feel that the subject is as much in the area of definability as of computability.[10]

Soare also rightly emphasizes that Turing (1936–37) gives an analysis of *human* computation, and states that only the later work of Gandy (1980) shows that machine-computable functions (which may involve parallel processing) are Turing computable.

In spite of the salutary and correct emphasis by Gandy (1980, 1988), Sieg (1994, 1997), and Soare (1996) that Turing's original paper was an analysis of *human* computation, and that for Turing, following his contemporary usage, "computer" means

a person who computes,[11, 12] not the later idea of a computing machine, nevertheless the influence of the "Turing machine" (and perhaps the subtle influence of his later philosophical paper Turing 1950) has led in many writings to a computer-science orientation to the problem. The contemporary situation, where high-speed computing machines are an essential part of our culture, is clearly also an influence.[13] I am, however, a logician and wish to explore the issue from a logical orientation. And here I have some genuine issues with what some other people appear to write.

The commonly used phrase "Turing's argument" suggests that Turing's 1936–37 paper gave a single argument for his analysis of human computability. In fact, as Shagrir has already mentioned at a conference in 2006,[14] Turing gave three in the original paper, one of which is the one generally remembered (the first; see section 1, 117, and argument I, 135–38). In many ways, the conventional emphasis on this argument is indeed appropriate, since it directly attempts to show, by a penetrating analysis of human computation and its limits, that anything humanly computable must be computable on his machine. (See also the exposition in Kleene 1952, 356–363, 376–381.) Nevertheless, I myself will primarily be concerned with his second argument (argument II, Turing 1936–37, 138) in the main body of this paper. But that will come later.

Soare, following Sieg (1994, 1997) and Gandy (1988), gives a careful mathematical and philosophical analysis of Turing's first argument. He proposes to break it into two different steps. One equates the informal notion of effective computability with what he calls "computorability," namely computability by an idealized human computer. Presumably this is a philosophical analysis. But then, Soare states (following Gandy 1988, 82; see also Gandy 1980, 124) that "Turing then proved *Turing's theorem: Any computorable function is Turing computable*. Although not proved in a formal system, Turing's proof is as rigorous as many in mathematics" (293).

As Gandy (1980, 124) stated Turing's theorem—"*what can be calculated by an abstract human being working in a routine way is computable*"[15]—it seemed to me to be too vague a statement to be regarded as a theorem without some further explanation. However, Soare's more detailed statement comes much closer, since he carefully divides the argument into a philosophical part, equating intuitive calculability with an analysis of what might be required of an idealized human computer ("computor"), and then a technical or mathematical part, showing that anything computable by such an idealized person is, in fact, Turing computable (i.e., computable on a Turing machine). Nevertheless, aside from the fact that, as Davis remarks, "this is a brilliant paper, but the reader should be warned that many of the technical details are incorrect as given" (Davis 1965, 115) the analysis contains at least one definite piece of handwaving. Turing (1936–37, 135) remarks that "in elementary arithmetic the two-dimensional character of the paper is sometimes used […] and I think that it will be agreed that the two-dimensional character of the paper is no

essential of computation. I assume then that the computation is carried out on one-dimensional paper, i.e. on a tape divided into squares." Maybe it would be agreed, but nothing has been argued, much less proved. Others have commented on this gap, and how it can be filled. Nevertheless, such considerations make it plainly desirable that the philosophical analysis of idealized human computability be built into an axiomatization, so that the argument can be properly evaluated. Soare's remarks already go in that direction, but an actual axiomatization has been given by Sieg (2008).[16]

Soare (1996, 296) proposes that "Turing's Thesis be used as a *definition* of a computable function as Turing and Gödel suggested." *Prima facie* this seems incompatible with his own assertion that Turing gave an elaborate proof of a *theorem* stating that his analysis of human computability gives the right notion. In the interesting section 3.4 (296–298), Soare argues that historically "other *theses* became definitions" [my italics], and quotes the arguments of other unnamed senior logicians who do not agree with his view.[17]

Did Turing regard his first argument, as Gandy, Soare, and perhaps Gödel before them, and many after them, have thought, as proving a theorem? If he did, why did he give two other arguments? In fact, he says: "All arguments which can be given [for the identification of the Turing-computable numbers with those that would intuitively be regarded as computable] are bound to be, fundamentally, appeals to intuition, and for this reason rather unsatisfactory mathematically." His first argument he calls "a direct appeal to intuition" (Turing 1936–37, 135).[18] It is certainly more than just that, since it gives an elaborate philosophical analysis of what an ideal computation might be. Nevertheless, Turing would be a very unusual case of a mathematician who proved a theorem without realizing that his own argument was such a proof.[19] My own alternative analysis will be very close to his second argument, which has mostly been ignored in discussions of Turing's 1936–37 paper. It will also be rather close to the argument given by Church (1936a).[20]

4.2 Computation as a Special Form of Mathematical Argument

My main point is this: a computation is a special form of mathematical argument. One is given a set of instructions, and the steps in the computation are supposed to follow—follow deductively—from the instructions as given. *So a computation is just another mathematical deduction, albeit one of a very specialized form.* In particular, the conclusion of the argument follows from the instructions as given and perhaps some well-known and not explicitly stated mathematical premises. I will assume that the computation is a deductive argument from a finite number of instructions, in analogy to Turing's emphasis on our finite capacity. It is in this sense, namely that I

am regarding computation as a special form of deduction, that I am saying I am advocating a logical orientation to the problem.

Now I shall state another thesis, which I shall call "Hilbert's thesis,"[21] namely, that the steps of any mathematical argument can be given in a language based on first-order logic (with identity).[22] The present argument can be regarded as either reducing Church's thesis to Hilbert's thesis, or alternatively as simply pointing out a theorem on all computations whose steps can be formalized in a first-order language.

Suppose one has any valid argument whose steps can be stated in a first-order language. It is an immediate consequence of the Gödel completeness theorem for first-order logic with identity that the premises of the argument can be formalized in any conventional formal system of first-order logic. Granted that the proof relation of such a system is recursive (computable), it immediately follows in the special case where one is computing a function (say, in the language of arithmetic) that the function must be recursive (Turing computable).

For example, if one has a valid argument from finitely many premises, and the natural number n is represented by $0^{(n)}$ (0 with n-strokes, i.e., successor symbols; a single function letter f might be the natural representation of successor), and the premises allow one to conclude for each particular n either $P(0^{(n)})$ or its negation,[23] it follows that the set defined by the predicate P is recursive (Turing computable). The same sort of argument will show that any function whose graph is computable in this sense by an argument whose steps are in a first-order language must also be recursive.

Particular examples are easily found from elementary primitive recursion. For example, which numbers are even can be decided from axioms stating that 0 is even, and that a successor of an even number is not even, while the successor of a number that is not even is even. One could similarly analyze the usual recursive definitions of addition, etc.

Although I do not find his argument II to be entirely clearly presented,[24] I take Turing to be arguing that precisely those sets which are computable in the sense of deducibility in first-order logic ("the Hilbert functional calculus" in his own terminology) are the Turing-computable sets (Turing 1936–37, 138–139). Church also, though not committed at that time, or at least in his initial paper, to any special primacy of conventional first-order logic, also argues that anything formalizable in a symbolic logic that is recursive (λ-definable), in its axioms and rules, must also compute only recursive functions. (See Church 1936a and the exposition in Sieg 1997.)[25] It is because of these considerations that I take myself to be reviving somewhat neglected arguments already given by Church, and especially Turing.

One should therefore notice the following: at the time the classical definitions of Church, Herbrand-Gödel-Kleene, and Turing were given, and in fact even rather

earlier, another definition of computability was at hand. One could have defined a set (relation, function) as decidable (computable) if instance-wise membership and non-membership in the set can be decided by a proof from a finite set of instructions (axioms) statable in a first-order language.[26] The point, as I have stressed, is that a computation is a deduction of a special sort. Given the Gödel completeness theorem, it follows immediately that this definition is equivalent to the classical definitions just mentioned, because intuitive provability can be identified with provability in a standard formal system. And these, of course, are systems with a recursive proof predicate in the classical senses just mentioned.

Taking the approach just mentioned as basic has advantages that I will go into shortly. The main advantage of any mathematical definition of computability, with an accompanying thesis identifying it with the intuitive notion, is that it allows one to prove negative results, that various sets are not decidable, etc. The present approach has that advantage in common with the others, but it is especially suited to the *Entscheidungsproblem*, as may already be clear, and will be spelled out below. There are also other advantages, some of a historical sort, also to be spelled out below.

Another relatively minor advantage of the approach of deducibility in a first-order language, but one which has bothered the present writer, is simply pedagogical. Ever since Post's famous paper (1944), the advantages of an intuitive presentation of computability arguments has been evident, rather than the rival approach using a formal definition of computability in the proofs. Although the experienced computability theorist will know how to convert such arguments into proofs using a formal definition, and cannot be said to be relying on an unproved thesis, this is hardly true for a beginner.[27] However, such a beginner, if he has already studied elementary logic, will readily accept that the steps of an argument can be stated in a first-order language, even if they are given verbally. The Gödel completeness theorem guarantees that if the steps really do follow (using any implicit axioms in addition to the actually stated steps), the argument can be formalized in one of the usual systems. Granted that the proof predicate of such a system is recursive/computable by one of the usual definitions, that one really has a technically valid proof will be readily accepted.

Soare remarks (see Soare 1996, 299) that Kleene's equation calculus and μ-recursive formalism, contrary to what some have claimed, is no better than the Turing machine or other formalisms in providing proofs that are readily comprehensible to the reader or student. He applauds the change to intuitive proofs (with reference to a machine formalism), as in Rogers (1967) and subsequent textbooks. Using the present approach, this change is easily justified by the definitions. (However, I cannot quite assert that all formalisms are impossibly difficult for the beginner, at least at an elementary level. In my own lectures [Kripke 1996], I had

some success using a formalism roughly equivalent to using Σ_1 definability for recursive enumerability. Students were supposed to give and understand proofs in the formalism, not invoking Church's thesis.[28] There may be other possibilities.)

For the reasons just given, I have thus proposed that derivability from a finite set of instructions statable in a first-order mathematical language be taken to be the basic technical concept of computability. However, if this is not done, one can use the Gödel completeness theorem for any conventional formalism to show the equivalence of this characterization to any standard characterization, by showing that the proofs in this conventional formalism are all decidable in the sense of the standard characterization. I take it to be the point of Turing's second argument for his own characterization in terms of Turing machines, that the equivalence of the characterization of computability in terms of formal first-order logic as just described and his own characterization in terms of Turing machines can be shown to be equivalent. His third argument appears to be a synthesis of the first two. As to the relation of the first two arguments, one can be thought as emphasizing human beings as operating mechanically on paper, the other as thinking beings, deducing the results of their computations from the instructions given.

The reference to Turing's second argument might be used to bring out two things. First, it would not matter if Turing had allowed two-dimensional or indeed n-dimensional computations, provided he had given conventional instructions for the operation of his n-dimensional machine at each stage. The argument would still follow as long as the operation of the machine was deterministically stated in conventional mathematical language, formalizable in first-order logic. The same holds if we view Sheperdson-Sturgis "register machines" as linear computations. The state of the machine at any stage is still a finite configuration that can be coded into a natural number.[29] One can then define a two-place predicate whose graph is a single valued function (or alternatively allow function letters; see below), $R(n, m)$, meaning that the nth stage of the computation is the state coded by m. Single valuedness can be an axiom, and the transition from a given stage to its successor can be given by statable axioms describing the operation of the machine. The value of the function being computed or predicate being decided can then be determined by appropriate axioms describing how it is to be retrieved from the final state. This approach is possible whenever we have a deterministic linear computation. So my second point is this characterization of arbitrary deterministic linear computations. Of course, one need not be so explicit, numbering the linear steps. One can simply present a particular computation, step by step, as linear.

However, one need not restrict oneself to linear computations. Kleene's formalism (Kleene 1952) with an equation calculus is an example of a nonlinear formalism. Generalizing this, one needs only a finite set of instructions for the computation, and the equations (or predicate assertions and their negations; see below) are

whatever can be deduced from these instructions by allowable rules. In such a case, the basic postulates may allow inconsistencies to be deduced (conflicting values for functions, or proofs of a formula and its negation). No real computation will allow such inconsistencies, but there will not in general be a syntactic criterion to rule them out. However, to obtain an enumeration theorem for all such computations, one will have to have a priority ordering for different deductions, with "least" deductions being preferred. This is, in effect, what Kleene does.

Some side remarks: Kleene pointed out that the formalism of recursion theory and the avoidability of a diagonal argument is best handled by making the theory a theory of "partial recursive" functions. If the formalism of first-order logic allows function letters and terms, it would thus be best to allow them to be partial. Given this, we might use free logic rather than traditional first-order logic, so that universal instantiation is allowed only in the presence of an existential premise. That is, the axiom scheme of universal instantiation becomes $((x)\ (\mathrm{F}x \wedge (\exists y)(y = t)) \supset \mathrm{F}t)$ (or in a natural deduction system the rule of universal instantiation is correspondingly modified; we assume the usual restrictions on substitution). In spite of its obvious connection with the theory of partial recursion, for some reason free logic has been of much more interest to philosophical logicians than to mathematical logicians.

Alternatively, one could not allow function letters and terms in the language, only relation symbols. Any function can be replaced by its graph. One can then use conventional first-order logic and not worry about the question of empty terms. Essentially, this was one of Russell's main ideas in his theory of descriptions (see Russell 1905 and Whitehead and Russell 1910).

Second, for the special type of mathematical argument I am considering (namely, computation), conventional first-order logic is clearly more than sufficient. Who would use in such an argument steps with premises involving many changes of quantifiers over an infinite domain? So, I think a really much more restricted language has to suffice here. (It would resemble Skolem arithmetic in allowing only bounded existential quantifiers.)[30] However, we do not need to worry about such restrictions. No argument needed here seems to be affected if we allow all first-order logic.

Also, when I refer to first-order logic, it is convenient in practice to allow many sorted systems, though one could reduce them to a conventional system with only one domain, and in my theoretical discussions immediately below, I will assume that this has been done.

Finally, though I believe, given the limitations of the human mind, that any actual computation is a derivation from a finite number of instructions, nothing is lost in terms of the actual power of the definition if infinitely many premises are allowed, provided these are already known to form a computable set (for example, a primitive recursive set of axioms).[31]

4.3 Von Neumann's Problem of Characterizing and Proving Unsolvability and Gödel's Theorem IX

Suppose we had taken derivability by a computation expressible in a first-order language as one's basic definition of computability. Then given the Gödel completeness theorem, any conventional formalism for first-order logic will be sufficient to formalize such derivability. In fact, in addition, one can give a universal characterization, also formalized in a first-order system, of derivability in arbitrary first-order systems. This will be the analog of the "universal Turing machine." It will be a short and direct step to conclude the undecidability in this sense of the *Entscheidungsproblem*.

Gandy (1988) gives a fascinating history of the development of notions of effective calculability.[32] He shows that the Hilbert school hoped and expected that the *Entscheidungsproblem* would prove to be solvable. It also hoped that such usual systems as Peano arithmetic would prove to be complete. The first expression of pessimism Gandy quotes within the Hilbert school[33] is from von Neumann (1927), who worries that a decision procedure for the *Entscheidungsproblem* would in effect abolish mathematics in place of a mechanical procedure, and therefore conjectures that such a procedure does not exist. He states, "We cannot at the moment prove this. We have no clue as to how such a proof of undecidability would go" (quoted in Gandy 1988, 62).

Gandy also has a somewhat similar quotation from G. H. Hardy:

> Suppose, for example, that we could have a finite system of rules which enabled us to say whether any given formula was demonstrable or not […]. There is of course no such theorem and this is very fortunate, since if there were we should have a mechanical set of rules for the solution of all mathematical problems, and our activities as mathematicians would come to an end. (Hardy 1929, 16; quoted in Gandy 1988, 62)

The context of this quotation is explicitly a discussion of Hilbert's metamathematics, and what would be desirable (a consistency proof) and undesirable (what we have quoted). However, he neither identifies this question with the *Entscheidungsproblem* for first-order logic, nor speculates (as von Neumann does) on the need for a proof of the impossibility of such a solution (or of such a finite set of rules).[34]

Gandy's next section discusses Gödel's Theorem IX in his famous paper (Gödel 1931; references are to the 1986 reprint). I think this theorem, and its anticipation of "Church's theorem," was for a long time neglected, but is now well known. The theorem states:

> *In any of the formal systems mentioned in Theorem VI, there are undecidable problems of the restricted functional calculus* (that is, formulas of the restricted functional calculus for which

neither validity nor the existence of a counterexample is provable). (Gödel 1931, 187, quoted in Gandy 1988, 63; italics in text, citations omitted)

Gödel goes on to say,

This is a consequence of Theorem X. *Every problem of the form (x) F(x) (with* [primitive] *recursive F) can be reduced to the question whether a certain formula of the restricted functional calculus is satisfiable* (that is, for every [primitive] recursive *F*, we can find a formula of the restricted functional calculus that is satisfiable if and only if $(x)\ F(x)$ is true). (Gödel 1931, 187)

Gandy goes on to comment as follows:

Since the systems considered are ω-consistent extensions of a formulation *P* of the simple theory of types, they are powerful enough to define and prove simple facts about the satisfaction relation for the predicate calculus. For any such system Σ [primitive recursive ω-consistent extensions of simple type theory] Gödel constructs a formula φ_Σ which is satisfiable, but for which this fact cannot be proved in Σ. As a consequence, given any proposed algorithm α for the Entscheidungsproblem and any system Σ, either it cannot be proved in Σ that α always gives an answer, or it cannot be proved in Σ that its answer is always correct. (Gandy 1988, 63)

He concludes: "Thus Gödel's result meant that it was almost inconceivable that the *Entscheidungsproblem* should be decidable: a solution could, so to speak, only work by magic" (63).

Gandy's conclusion here strikes me as much too weak. It seems to me that Gödel's result *proves* that the algorithm α simply *cannot* be correct. First, we can simply suppose that in our extended system Σ we include axioms describing the alleged algorithm α. It describes a computation, which according to my familiar viewpoint, if it is to be an appropriate algorithm, its steps should follow from each other, and therefore be provable given the Gödel completeness theorem.[35] Let $A(x)$ be the statement that α terminates with the conclusion that *x* satisfies the algorithm. Then for each *n*, either $A(0^{(n)})$ or its negation is provable.[36] If the algorithm α is supposed to coincide with validity, simply add an axiom saying so, $(x)(A(x) \equiv \text{Val}\ (x))$. The resulting system is not ω-consistent, and therefore its axioms cannot be true. Hence (assuming the fault does not lie in the underlying system), the additional axioms cannot be true: α is supposed to be a genuine algorithm whose steps really follow from each other, so the fault can only lie in the failure of the last axiom $(x)(A(x) \equiv \text{Val}\ (x))$ to be true.

Note that though Gödel stated his result for extensions of simple type theory, he knew and claimed that it held for a variety of primitive recursively axiomatizable extensions of various systems, ranging from ZF set theory to weaker systems, such as first-order number theory.[37]

Another way of looking at the matter of Gödel's Theorem IX is as follows. In the same paper, Gödel introduces the notion of an *Entscheidungsdefinit* predicate[38] (in modern terminology, one that is strongly representable, binumerable, or numeralwise decidable), and states that his basic undecidability result (Theorem VI) still holds for any *Entscheidungsdefinit* ω-consistent extension of the basic system, since this is the only property of primitive recursiveness that has actually been used. He then states that the predicate of the universal statement that is undecidable would have to be *Entscheidungsdefinit*, not necessarily primitive recursive. A decision procedure for the predicate calculus would lead to an *Entscheidungsdefinit* predicate for validity or provability in some basic extension of the axiom system.

Nowadays, we would know that the form of the undecidable statement would not really change, and could go on to Gödel's Theorem IX as before. However, Gödel does not seem to have known this in 1931. In fact, he does not realize until 1936[39] (about the same time that the work of Church and Turing appeared) that for all reasonably strong systems, the class of sets definable by an *Entscheidungsdefinit* predicate is the same (the recursive sets).

Gandy goes on to state that Gödel's work did lead to skepticism about the solvability of the *Entscheidungsproblem* (63, section 6.1), and specifically mentions Herbrand and Schütte. In section 6.2, he asks, "A natural question to ask is why neither Gödel nor von Neumann *proved* the undecidability of the Entscheidungsproblem" (63). Gödel did not seem to have *regarded* himself as proving the undecidability. In his later paper (Gödel 1933), he gives a decidable class ("the Gödel case") and then shows that what may be natural to regard as the next largest case is a reduction class; that is, a decision procedure for it would lead to the decidability for the whole predicate calculus.[40] He does not say that such a decision procedure has been shown to be impossible. And we all know that he felt the question to be decided only by Turing's work.

From conversation I have the impression that some logicians feel that Gödel came very close to a negative solution to the *Entscheidungsproblem*, or even that all that was lacking was a version of Church's thesis. In any event, I feel that this is so. A decision procedure for the *Entscheidungsproblem* would have to be a procedure statable in the ordinary language of mathematics, whether the procedure operates by a machine program or instructions for a human computation. (Notice that in Gandy's exposition, ending with a weaker claim, it is still assumed that the algorithm α is statable in some extension of P, or whatever basic system is involved.) It could presumably be written in the language of P, perhaps with additional predicates and an additional finite number of instructions. Gödel's Theorem IX clearly directly implies Turing's result that the *Entscheidungsproblem*

is not decidable on one of his machines, since we can simply add an axiomatization of the operation of the machine to his basic system.[41] As I said, the result would still obtain if Turing had allowed many dimensions. Similarly for a large class of mechanical procedures.

Thus I regard Gödel's Theorem IX as already answering von Neumann's 1927 problem, "we have no clue as to how such a proof of undecidability would go" (quoted by Gandy 1988, 62). In this respect, I disagree with the presupposition of Gandy's question quoted before: "A natural question to ask is why neither Gödel nor von Neumann *proved* the undecidability of the Entscheidungsproblem" (63). But Gandy's question is appropriate in the following form: why didn't Gödel and von Neumann regard Theorem IX as such a proof? One problem in the argument I have given that Theorem IX is such a proof is its free use of the notion of truth, and locating the trouble in one extra axiom that must be false. However, it seems very unlikely that Gödel, at least, would have regarded that as a questionable part of the argument.[42] What seems most likely lacking is an appropriate analog of Church's thesis.

4.4 Some Clarificatory Remarks on the Present Characterization

Let me state a few issues that I exclude from the domain of the present characterization of computability. In Gödel (1972), Gödel finds what he regards as "a philosophical error in Turing's work" (306). It is not utterly clear to me what Turing's error is supposed to be, but it is characterized as "an argument which is supposed to show that mental procedures cannot go beyond mechanical procedures" (306).[43] The error appears to be in Turing's assumption that only a finite number of states of mind need to be admitted. He says:

> What Turing disregards completely is the fact that *mind, in its use, is not static, but constantly developing* […]. 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. Note that something like this indeed seems to happen in the process of forming stronger and stronger axioms of infinity in set theory. This process, however, today is far from being sufficiently understood to form a well-defined procedure. It must be admitted that the 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.[44] (306)

Arguments by Gödel cannot be taken lightly, but it is difficult for me to see how Gödel has found any philosophical error in Turing's work, if that work is supposed to show what number theoretic functions can be regarded as effectively computable, even though we do not identify mental calculation with mechanical or routine cal-

culation. Gödel directs his argument against Turing's argument I, but the objection, if valid, would apply to my own approach also, similar to Turing's argument II. It is difficult for the present writer to see why Kleene is not obviously correct when he says,

> Thus algorithms have been procedures that mathematicians can describe completely to one another *in advance* of their application for various choices of the arguments. How could someone describe completely to me *in a finite interview* a process for finding the values of a number-theoretic function, the execution of which process for various arguments would be keyed to more than the *finite* subset of our mental states that would have developed by the end of the interview, though the total number of our mental states might converge to infinity if we were immortal (Kleene 1988, 50).

Thus I have regarded a function as computable when its values can be obtained for all arguments from some fixed finite set of instructions.[45]

It does not seem to me particularly relevant that the directions be public, that more than one person is involved.[46] It suffices that someone gives a finite set of directions to herself. Also it does not matter that the calculation be "mechanical," if this is a genuine limitation.[47] Any intelligible set of directions suffices, as long as they can be understood in classical first-order logic.

I also exclude from my discussion any consideration of computability based on intuitionism or intuitionistic ideas, in particular on the "intended interpretation" of intuitionistic logic. Kreisel has emphasized the following: if one has a proof of a statement of the existence of a natural number in an intuitionistic formal system, then, if intuitionism is really constructive, one must be able to calculate from the Gödel number of the proof of the existential statement to the numerical instance of that statement *given* by the proof (Kreisel 1970). And notice that though we may have various technical proofs that a particular intuitionistic system has "the numerical instantiation property" and some such technical proof may determine an appropriate recursive or Turing-computable function, who knows whether that is the "right" number in Kreisel's sense?[48] Certainly, Turing's analysis of computation (his argument I) does look irrelevant here, as does that of the present paper, based on classical first-order logic. I simply wish to exclude this problem; intuitionism plays no role in the sense of computability developed here. I think this sense includes anything called a "calculation" in the ordinary sense. (This sense is also developed in Turing's argument I.)

I also wish to exclude questions of empirically or physically based computation (which has already been alluded to in Martin Davis's talk at a 2006 conference). Below I will give an example of what I wish to exclude. Here we are talking about *mathematical* computation, whether done by a human being, or a machine,[49] or anything else.

Robin Gandy wrote a well-known and mathematically ingenious paper trying to generalize Turing's first argument to computing machines (Gandy 1980). He reports that many with whom he discussed the matter doubt that there is any problem to be solved, but others agree with him that Turing's analysis might fail for modern computing systems that use parallel processing. In his paper, Gandy gives a formal model (in hereditary finite sets from a set of atoms) for such computation. He then imposes a locality condition, which is supposed to mean that although parallel processing is allowed, one branch cannot influence another that is too far apart.

My main point in this part of the discussion will be to propose the basic approach of the present paper as an alternative to Gandy's, not only for human computation but for machine computation, including the types of cellular automata that worried Gandy. But no claim will be made about what machines are empirically realizable.

If Gandy had stipulatively defined his class of machines, there would be no issue of an empirical basis.[50] However, in fact Gandy appeals to special relativity to justify his new locality condition. (The idea is that no influence can travel faster than light.) He even wonders whether a machine whose operation is compatible with Newtonian physics might compute a nonrecursive function, even an arbitrary function (in his colorful terminology, exercise "free will").[51]

Some important writers have held that Gandy was indeed the first to prove that actual digital computers do compute only the recursive (Turing-computable) functions.[52] However, an appeal to special relativity makes the adequacy of Gandy's analysis an empirical issue, depending on the actual state of physics. Because of this a wealth of literature gives putative counterexamples, in both hypothetical[53] and possibly true physics (see Copeland and Shagrir 2007 for a survey). There appears to be a particular interest in what I have been inclined to call "Zenonian calculation," where one can perform an infinite number of operations in a finite time, analogously to Zeno's paradox. Then one could plainly go beyond the recursive or Turing-computable functions, as Copeland and Shagrir explain.[54] In particular, many authors are discussed who speculate on models of general relativity in which what appears to be a computation of the infinite (and thus maybe Zenonian) type in one part of the model is only finite in time from another, later point of view. This literature is only slightly known to me, and much is beyond my competence to evaluate, would I have studied it. However, it does seem clear that if one bases Gandy's thesis on empirical questions of physics, one is on dangerous ground, even if there is no known violation by the computing machines of today.

Copeland and Shagrir also do discuss (2007, section 3) "non-discrete machines" that make use of continuously valued quantities. As they say, in Gandy's published paper he explicitly excludes "devices which are *essentially* analogue machines" (Gandy 1980, 125), though he mentions some possibilities. In unpublished work,

Gandy does consider such devices, and conjectures a strategy to succeed in showing that no feasible such device goes beyond Turing computability (see the discussion of Gandy's unpublished manuscript in Copeland and Shagrir 2007, section 3).

One very simple example of that kind that has always troubled me is as follows. Consider such dimensionless physical constants as the electron-proton mass ratio, or the fine structure constant.[55] As far as I have heard, at the present time physical theory has nothing much to say about the mathematical properties of these numbers, whether they are algebraic or transcendental, or even rational or irrational. In particular, it might well be the case that the decimal expansions of these numbers are not Turing computable. Assume this is so, and also assume that time is infinite in extent and that there are no limitations in energy that prevent computations of these decimals to any number of places. As far as I can see, nothing in principle rules this out. Then an empirically possible machine would exist that calculates a decimal not computable in Turing's sense.[56, 57]

This example could have philosophical implications for the physical applicability of other theories. Hermann Weyl, in his famous monograph (Weyl 1918), showed how a great deal of analysis can be done allowing only arithmetically definable reals. Extensive and mathematically impressive work, which I much admire, has been done by Solomon Feferman, extending Weyl's ideas to richer systems with higher types. Feferman emphasizes that everything can be done in a conservative extension of Peano arithmetic (first-order arithmetic). Here we are concerned only with the question of whether this work is adequate for physics, which Feferman has explicitly claimed to be the case (see Feferman 1992).[58]

Not only is there no reason to suppose that the electron-proton mass ratio, described as a decimal, is Turing computable, there is no reason to suppose that it is arithmetical, hyperarithmetical, or, for that matter, definable in set theory. For all we know, the physical theory of the future may somehow imply that the truth set for Peano arithmetic is definable in terms of the electron-proton mass ratio, and even use it to show that physical theory is not a conservative extension of that system.

Let me propose an alternative to Gandy's approach, divorcing the question of machine computability from all empirical assumptions. As I said, computation is simply a form of mathematical argument. Let us consider only those devices that are describable in a first-order language, and whose program is such that the successive states logically follow from each other, one by one, together with the program and perhaps some basic mathematical assumptions. Any particular computation by the machine is assumed to be finite in length, and the machine states describable finitely, and following each other discretely. This I wish to assume whether or not parallel processing is involved. Certainly, this class of machines includes Conway's Game of Life (see Gardner 1970), which Gandy specifically cites as much of the inspiration for his paper.[59, 60, 61]

Let me now return to our basic topic. Church's thesis, in its weakest form, is simply that the decidable sets are recursive, or correspondingly for functions and partial functions. Thus if we know that a set of natural numbers is finite, say by knowing an upper bound to the set, we know that the set is recursive, even though we may have in fact no knowledge which numbers are in the set and which are not.[62]

Thus there is some interest in what might be called an "intensional," or alternatively, "effective," form of Church's thesis. The term "Church's superthesis" has also been used (I think it is due to Kreisel). The idea is that any intuitive set of instructions for a computation can effectively be transformed into a computation given in one of the conventional formalisms (Post's "routine chore"). Philosophically, this strikes me as interesting for the following reason. The usual formalisms for Church-Turing computability transform any argument using the intuitive notion of computability on the natural numbers into one using a chosen formal notion. The intuitive argument can disappear as part of the motivation but not of the argument itself, which can be formalized in set theory, or indeed in number theory. However, this is no longer so for the present case. A significant effectiveness claim is being made, but it is hard to see how it can be replaced by a formal claim, statable in set theory. Nor is it an intuitionistic claim; it is supposed to be directly intelligible to the classical mathematician, and may apply to intuitive effectiveness arguments that require further work to be valid intuitionistically (perhaps it cannot even be made intuitionistically valid). Kleene's S_n^m theorem could be considered a partial formalization of this claim, but not completely so.[63]

If we are given a linear computation, in terms of the present formalism, the problem is simply writing out the steps explicitly in the formalism of first-order logic. The completeness theorem, given that the steps follow from each other, will guarantee that the steps follow in a usual formalism for first-order logic.[64] As I already mentioned from the pedagogical point of view, this is easier than translating into one of the other standard formalisms.

The translation is of course more natural if it is done in a natural deduction system rather than in a Hilbert-type system. Nevertheless, there remains a further problem, also to be stated in terms of the effective form of Church's thesis. For so far, we have shown only that each step of a (discrete) linear computation will follow from its predecessor and the appropriate instructions. However, we have not shown that a formalized deduction will not need irrelevant steps in between the formalized intuitive steps, steps that would not have been used in the intuitive argument. Here Prawitz (1965) is a monograph that would seem to be specifically relevant, since it shows how to eliminate unnecessary steps in a natural deduction argument.[65] (Fine 1985 may also be relevant.)

Although I am trying as much as possible to appeal to actual theorems, in this case to show that the computation can be formalized without unnecessary steps not

to be found in the intuitive argument, I certainly cannot claim to have thought this aspect of the matter through. The work of Prawitz (1965) may not always be the right thing to cite because the intuitive computation being formalized may itself contain unnecessary steps that would have been eliminated by a clever researcher. Or, perhaps, one is dealing with a procedure that perversely goes about things in a roundabout way. As in other cases of translating recursion theoretic arguments into a formalism, but here more basically as a special case of the formalization of mathematical arguments, it may be that we should best appeal to practical experience, antedating Gödel's completeness theorem, let alone the more sophisticated work just mentioned. That in practice such arguments can be so formalized already goes back to the early work of Frege, Russell, et al., in formalizing mathematical arguments. Today this would be most naturally done in a natural deduction system. But the steps in the formalization would by definition mirror those in the informal argument. Nevertheless, it would be the Gödel completeness theorem that guarantees that there is not some intuitively valid argument not formalizable in the usual formalisms.

4.5 Conclusion

How far does this paper go toward satisfying the expressed desire of some people (including my earlier self) for a proof of Church's thesis? The basic idea, assuming a set of intuitively evident axioms for computability, is most explicitly given in two contemporary papers, several by Sieg (see Sieg 2008, and other writings), and Dershowitz and Gurevich (2008).

Arguably, the present argument, which does not give a particular axiomatization,[66] but does simply propose a new characterization of computability, only reduces one unproved hypothesis to another one, namely, that the statements in mathematical arguments can be described in first-order systems. It then points out that the formal characterization of computability can be derived from the Gödel completeness theorem (in fact, that the formal characterization might well be given directly in terms of first-order formalizability).

However, surely something is achieved here. Even the form in which I have just stated it, reducing one thesis to another one, is a fact worth noting. And various consequences of this fact have been noted above. But also, we can state a theorem restricted to algorithms whose steps can be stated in first-order logic, and the special case of the Gödel completeness theorem can be called the "first-order algorithm theorem." I took this term to be fundamental in earlier versions of this paper. First-order algorithms happen to include all known algorithms, whether done by human or machine.

Some of the authors I have discussed also let their results depend on the primacy of computations whose steps are statable in first-order logic. Although in its discussion of his machines, Gandy (1980) makes the development entirely independent of any reference to first-order logic, in its discussion of Gödel's Theorem IX (as quoted above), Gandy (1988) does not question that the algorithm α in question should be formalizable (statable) in some suitable axiomatizable extension of the basic system *P* (or whatever), perhaps with new predicates. On this basis, I have argued that his conclusion about Theorem IX ought to have been stronger. More basically, the highly ambitious paper by Dershowitz and Gurevich, giving an axiomatization leading to a proof of Church's thesis that they think is more basic and general than earlier ones (such as are explicit in Sieg and implicit in Gandy et al.), makes a fundamental appeal to the notion of an arbitrary first-order structure. They write: "The justification for postulating that structures or algebras are appropriate for capturing algorithmic states is the enormous experience of mathematicians who have faithfully and transparently presented every kind of static mathematical reality as a first-order structure" (Dershowitz and Gurevich 2008, 317). They rightly state (and see my corresponding discussion above) that their notion of structure is more general than Gandy's idea of a machine as represented as a hereditary finite set, and mention numerous other computing languages, etc. So they appeal to the same mathematical experience that I do.[67]

So, to restate my central thesis: computation is a special form of deduction. If we restrict ourselves to algorithms whose instructions and steps can be stated in a first-order language (first-order algorithms), and these include all algorithms currently known, the Church-Turing characterization of the class of computable functions can be represented as a special corollary of the Gödel completeness theorem.[68,69]

Notes

1. The present paper is an edited transcript of a talk delivered at the 21st International Workshop on the History and Philosophy of Science, "The Origins and Nature of Computation," held in Tel Aviv and Jerusalem, Israel, on June 12–15, 2006. The fact that this is a transcript of a paper delivered orally, rather than written, accounts for a certain amount of conversational tone. A version of this lecture was first given in August 1998 at the Association for Symbolic Logic in conjunction with the International Congress of Mathematicians, Humboldt University, Berlin, Germany.

2. See the explicit statement on 318: "While we cannot prove Church's thesis, since its role is to delimit precisely an hitherto vaguely conceived totality, we require evidence that it cannot conflict with the intuitive notion which it is supposed to complete." Kleene's section 62 is a summary of the evidence. Much of it is presented in detail in other sections. For another statement of the conventional view, see, for example, Cutland (1980). He says: "Note immediately that this thesis [Church's thesis] is not a *theorem* which is susceptible to mathematical proof; it has the status of a *claim* or *belief* which must be substantiated by evidence" (67). Cutland follows this statement with a brief survey of the evidence.

3. Dershowitz and Gurevich (2008, 305), who quote Shoenfield's later statement, also refer to a letter from Church to Kleene, quoted by Davis (1982, 9). This letter, dated November 29, 1935, does quote

Gödel as suggesting "that it might be possible, in terms of effective calculability as an undefined notion, to state a set of axioms which would embody the generally accepted properties of this notion, and to do something on that basis" (Davis 1982, 9). This would indeed be a striking anticipation at a very early date, by the greatest logician of all time, of the later writers I have referred to. However, the full context quoted makes this anticipation a bit weaker. He was fending off a challenge from Church to give a mathematically defined class of effectively computable functions that he (Church) could not prove to be λ-definable. Certainly, he didn't publish this remark.

Nevertheless, by the time my lecture was given in Israel perhaps the number of authors who agree with my view was great enough that I shouldn't have referred to the opposite idea (that no mathematical proof of Church's thesis could possibly be given) as the "conventional view." I should add that both Soare (1996) and Gandy (1980, 1988) think that Turing (1936–37) already gave a mathematical proof that any function calculable by a human being in a routine way was Turing computable, as opposed to simply a convincing argument for this thesis. That Turing himself seems to have held the second and more modest view is argued in my main text. (All future page references to Turing 1936–37 are to the 1965 reprint.)

4. In my (1996) manuscript *Elementary Recursion Theory and Its Applications to Formal Systems*, I express my agreement with the view of Shoenfield (1993) and Harvey Friedman that the equation of mathematical recursiveness and intuitive computability is a meaningful mathematical problem. I even give a proof in one direction that the mathematical concept implies the intuitive one (see next note). However, I state that "while this [a rigorous proof that the intuitive and the mathematical concept are the same] is in principle possible, it has not yet been done (and it seems to be a very difficult task)" (14). See also the references in Dershowitz and Gurevich (2008, 339), quoting me to that effect and Richard Shore in agreement with me. However, Friedman is quoted as more hopeful that reasonably soon Church's thesis will be proved in the appropriate sense (using weak and intuitive axioms for computation).

5. I give my own proof of this (easy) half in my manuscript Kripke (1996, 14–15). Actually, what I proved is that every recursively enumerable relation (or set) is semi-computable (i.e., has a proof procedure). I then characterized a relation as computable when it and its complement are semi-computable. Note my further comments that this is a mathematical proof that cannot be formalized in ZF set theory, so that if the statement that "all mathematics can be formalized in set theory" is taken too strongly, it is not so. The view is sometimes taken that the problem ought to be regarded as one in intuitionistic mathematics. This is not the point of view taken in this version of the argument. Indeed, intuitionistically the characterization of a decidable relation as one such that it and its complement are semi-computable is not obvious, and requires supplementary argument. See also my stipulative exclusion of using intuitionistic ideas in the notion of computation in the text below.

6. In his famous paper, Gödel (1931), Gödel actually calls them "the recursive functions." The term "primitive recursive" stems from Kleene (1936). See also Kleene (1981, 53).

7. I find that Barendregt (1997, 187) also gives the same argument I have just given about the Ackermann function. In fairness to Barendregt, and contrary to what I argue immediately above, his position is that, like the thesis on primitive recursiveness, Church's thesis could be disproved, but it cannot be proved. Our failure to find a similar disproof is thought to be empirical evidence that the thesis is true. On the other hand, Dershowitz and Gurevich (2008, 304) quote Barendregt's argument in the context of a paper whose very title, "A Natural Axiomatization of Computability and Proof of Church's Thesis," indicates their belief that Church's thesis can be proved.

In further fairness to historical accuracy, I should add that giving a counterexample to an identification of the primitive recursive functions with the effectively calculable functions does not appear to have been Ackermann's main goal in the paper.

Today, independently of Ackermann's work, it would be obvious to us that primitive recursion could not exhaust intuitive effective calculability. We can effectively enumerate all directions for computing primitive recursive functions, and then use a diagonal argument to define a calculable function that cannot be primitive recursive. On the basis of Kleene's discussion of the matter I credit this argument as originally given by Péter (1935). Kleene also states that Péter in the same paper (and also Robinson 1948) simplify Ackermann's original example (see Kleene 1952, 271–272).

I might make one remark about the argument I have given. Certainly the existence of a measurable cardinal could be disproved in ZF set theory, and in fact this has been attempted. However, the existence of a measurable (and other large cardinals) cannot be proved in ZF set theory, and I do not detect any

particular interest in finding evident seeming extra axioms that might favor their existence. Advocates of large cardinals seem rather to favor them because of their fecundity in proving allegedly plausible consequences. Note that it is still true that, if one believes in conventional ZF set theory, the existence of a measurable cardinal is of course a definite mathematical question.

8. I should mention that Soare (1996) distinguishes between Church's thesis or argument, which he regards as defective, and Turing's thesis or argument as intensionally different, and only the latter as really convincing. Following conventional terminology, I am not worrying about this distinction here, since the "theses" have been proved equivalent.

9. Soare also quotes Davis as telling him that Gödel reacted sharply to the term "recursion theory" for the subject, and that the term should be "used by reference to the kind of work Rózsa Péter did" (307–308). Nevertheless, there are places in print in the Gödel papers themselves where Gödel uses such terms as "general recursiveness (or Turing's computability)" (Gödel 1990, 150) and "a non-recursive number theoretic function" (Gödel 1990, 306). I don't know if this shows very much.

Soare's proposal has been generally adopted, so that terms such as "recursively enumerable" have given way to "computably enumerable." Agreement with Soare has not been unanimous among writers in the subject. I have been told that Martin Davis sees little reason for the change. Slaman, quoted in the text, also dissents from the view.

10. I heard Slaman advocate the view that the subject should be a theory of definability as much as computability in his Gödel lecture at the 2001 meeting of the Association for Symbolic Logic. The title of the lecture is "Recursion Theory."

In my own manuscript Kripke (1996), I independently had taken a definability approach to the subject myself. In fairness to Soare, he states "other concepts [as opposed to computability] (recursion, definability) are very important but not at the center" (Soare 1996, 314), thus explicitly rejecting the view of the subject taken by Slaman and myself.

Looking at this matter again now, I find that Soare may well be correct as to what has become the most important concept. However, like Slaman, I find definability a very important concept (as Soare acknowledges) and important for a good development of the subject and for its applicability to formal systems.

11. Even in much more recent (late-twentieth-century) dictionaries I happen to possess, the definition of a "computer" as such a person is still recognized as one possible meaning. It is actually hard to imagine that it could be otherwise.

12. In these papers the authors propose to adopt the spelling "computor" for Turing's notion of a human being making a computation, so as to avoid confusion with the contemporary notion of a "computer" as a machine. It is even proposed to take Turing's paper as giving a characterization of the "computorable" numbers (functions), again to avoid confusion.

Sieg (1994, 71) quotes Wittgenstein: "Turing's 'Machines'. These machines are *humans* who calculate" (Wittgenstein 1980, § 1096). The quotation, though insightful, is somewhat confusingly put. Better would have been: These machines are Turing's mechanical model of *humans* who calculate.

13. See Shagrir (2002), who argues that whatever the historical situation about Turing's original analysis may be, "Today computer scientists view effective computability in terms of finite machine computation" (221, abstract). I myself recall the lecture notes from one logician who stated the evidence for Turing's thesis in terms of contemporary computing machines. He emphasized that every high-speed computer (idealized appropriately in infinite capacity and in terms of its programming language) is in effect a universal Turing machine. Shagrir gives various other examples from contemporary writings. All or most seem to be written independently of Gandy's worries about the distinction between Turing's original analysis and goals and the machine formulation of the problem.

Yet another influence is Turing's own historical role in the development of high-speed computing devices.

14. Shagrir (2006) refers to both argument I and argument III.

15. To understand Gandy, it ought to be explained that "working in a routine way" is meant to exclude Gödel's objection to Turing's (first) argument, based on the notion that the mind might provide creative insights enabling it to go beyond Turing's analysis. He refers to Wang (1974, 325–26); I will discuss this objection in the main text below.

16. Although Soare is right to state that a rigorous mathematical argument need not be presented in a formal system, once one sees the desirability of an informal axiomatization, in the spirit of other con-

temporary axiomatizations, the rest would only be a routine matter of formalization, though it is not necessary.

17. This discussion raises a fascinating issue in the history and philosophy of mathematics and deserves considerable discussion that cannot be made here.

But to one part of the discussion I would like to make one correction (as I see it). Everyone relies on Kline: "Up to about 1650 no one believed that the length of a curve could equal exactly the length of a line. In fact, in the second book of *La Geometrie*, Descartes says the relation between curved lines and straight lines is not nor ever can be known" (Kline 1972, 354; quoted by Soare 1996, 297). But all this strikes me as very unclear. Descartes does say this. But to say that the relation between curved and straight lines can never be known is not clearly to be construed as claiming that the length of a curve and that of a line must be different, only that they are forever undecided. One might as well take a statement that the relation between the power of the continuum and $\aleph_1$ will never be known as asserting categorically that they are different. (Greater study of Descartes than I have made is needed, but I would presume that Descartes was uncertain whether the geometric straight line is complete. Otherwise, one would have to assume that he was unaware that the circumference of a circle can be arbitrarily approximated by inscribed polygons, and of the way π is calculated.) In addition, Kline presents no evidence that until 1650 everyone assumed that Descartes was right. So here I think Kline has gone considerably beyond his stated evidence.

As again Soare's proposal that Turing computability be taken as a definition of computability: "One senior logician objected to this proposed definition because he said we should view the Church-Turing Thesis as certainly correct, but as 'a one of a kind, without any true analogue in mathematics. I think we recursion theorists [*sic*; note the terminology] should be proud of this, and not (as you seem to suggest) replace it by a change of our definitions.' There is no reason why we cannot use Turing's Thesis as a definition of computability and still maintain awe and pride at a fundamental discovery" (Soare 1996, 296). But Soare regards Turing's argument I as the proof of a *theorem*. Does he give any example where a *theorem* (as opposed to a thesis or analysis *simpliciter*) became a definition? (He proposes other "theses" that, according to him, became definitions.)

18. That Turing says this is pointed out by Sieg (1994, 97). Paradoxically, this is the very same author who has done more than anyone else to justify the later claims that in his first argument Turing proved a theorem, by giving an axiomatic development of his work (see Sieg 2008).

19. Nevertheless, it is of course possible that Turing was excessively modest in his claims, and that he had, in effect, proved a theorem.

Moreover, I myself argue below that Gödel could have regarded his Theorem IX (Gödel 1931) as a proof of the undecidability of the *Entscheidungsproblem*, but did not do so. For this case, see the discussion below.

20. To put the matter more precisely, Turing's second argument presupposes some viewpoint such as the one taken in the present paper. All references to Turing's paper are to the 1965 reprint. See also the exposition, history, and analysis by Sieg (1997).

In addition to crediting Turing and Church themselves, I should add that I have heard reports to the effect that Martin Davis gave some consideration to a justification of Church's thesis in terms of first-order logic some time ago. At the Berlin conference (see note 1, this paper) Ronald Jensen told me that in lectures he had always emphasized formalizability in first-order logic, but had not at that time believed there was a theorem to be proved.

21. Martin Davis originated the term "Hilbert's thesis"; see Barwise (1977, 41). Davis's formulation of Hilbert's thesis, as stated by Barwise, is that "the informal notion of *provable* used in mathematics is made precise by the formal notion *provable in first-order logic*" (Barwise 1977, 41). The version stated here, however, is weaker. Rather than referring to provability, it is simply that any mathematical *statement* can be formulated in a first-order language. Thus, it is about statability, rather than provability. For the purposes of the present paper, it could be restricted to steps of a computation.

Very possibly the weaker thesis about statability might have originally been intended. Certainly Hilbert and Ackermann's famous textbook (Hilbert and Ackermann 1928) still regards the completeness of conventional predicate logic as an open problem, unaware of the significance of the work already done in that direction. Had Gödel not solved the problem in the affirmative a stronger formalism would have been necessary, or conceivably no complete system would have been possible. It is true, however, that Hilbert's program for interpreting proofs with ε-symbols presupposed a predicate calculus of the usual form. There was of course "heuristic" evidence that such a system was adequate, given the experience since Frege, Whitehead and Russell, and others.

Note also that Hilbert and Ackermann do present the "restricted calculus," as they call it, as a fragment of the second-order calculus, and ultimately of the logic of order ω. However, they seem to identify even the second-order calculus with set theory, and mention the paradoxes. Little depends on these exact historical points.

22. Throughout this paper I will mean first-order logic with identity when I refer to first-order logic. Other qualifications and explanations will be given below in the text.

23. But of course not both. A proper computation does not give inconsistent instructions.

24. But the fault may be mine in concentration rather than Turing's in exposition. However, see also the correction to the treatment of the unsolvability of the *Entscheidungsproblem* in Turing (1936–37, 152–153), which Turing attributes to Bernays.

25. According to the account in Sieg (1997)—and, of course, see also Church (1936a)—Church originally thought (in what has been called his "step by step" argument) that if a formal system did not have recursive rules, they would have to be unrecognizably complicated, and it would be doubtful that they could be effective. He also emphasized that his requirements were satisfied by all existing systems of symbolic logic.

All charges of circularity against Church would disappear if one supplemented his argument with a recognition of the primacy of first-order logic (or at least in the metatheory of the formal system described), and Gödel's completeness theorem for an appropriate set of rules for first-order logic. Church's famous paper (1936b) can be regarded as an implicit recognition of this primacy. And his argument is certainly that computation is a form of deduction within a system, very much in the spirit of the present paper.

26. And for sets, similarly for relations and functions. Also, one can regard a set as admitting a proof procedure (or semi-computable) if such a procedure allows us to prove membership in the set if and only if it holds, but says nothing about nonmembership (the analog of what is classically called recursive enumerability). It is of course a classical result of recursion theory (computability theory) that this is a weaker notion than decidability.

27. See Cutland (1980). He says: "For the practiced student there is the additional evidence of his own experience in translating informal algorithms into formal counterparts. For the beginner, our use of Church's thesis in subsequent chapters may call on his willingness to place confidence in the ability of others until self confidence is developed" (70). The trouble is that someone who learns computability theory from a text such as Cutland's probably will never have developed such confidence by himself and will forever rely on others. One who learns the basics from a book such as Kleene (1952) may get lost in the formalism.

28. Some basic theorems, in particular the enumeration theorem for recursively enumerable sets, may be simpler using such a formalism, or one of the traditional formalisms, than the proposed formalism based on logic. However, the logical formalism will have its own version of the universal Turing machine in a formulation in first-order logic of all first-order formalisms. See the text below.

29. Such coding is not really necessary. The states could be directly represented in the first-order language. No doubt the natural numbers, or *n*-tuples of them, would be natural for representing the individual cells of the machine.

30. The Herbrand-Gödel-Kleene equation–calculus formalism somewhat resembles such a narrow system, but is differently motivated, as stated.

31. Of course, many or most of the conventional systems of mathematical logic have infinitely many axioms above those of logic itself; for example, the usual first-order arithmetic, or the usual formalization of ZF set theory. The philosophical difficulties involved in our ability to understand these systems are interestingly discussed in Martin's paper "Sets versus Classes" (1974), which has circulated but is unpublished.

32. Another interesting and later history and analysis is given by Sieg (1994), and has been used above.

33. For the full quotation from von Neumann, which is very interesting, see Gandy (1988, 61–62). Outside the Hilbert school, Gandy discusses Brouwer's skepticism as to the decidability of all mathematical problems and Post's claim to have early shown the existence of a recursively enumerable set that is not recursive (61). Gandy mentions (62) that in fact mechanizability in principle would not necessarily put mathematics at an end. For example, some problems in elementary geometry that in principle might be

solved by Tarski's decision procedure still are unsolved and require conventional mathematical methods in practice.

34. Gandy writes: "Hardy might have read von Neumann's paper, or have heard of it from F. P. Ramsey; in his lecture he discusses Hilbert's ideas at length, but he does not refer to von Neumann" (Gandy 1988, 63). Actually, Gandy is mistaken on this factual point. Hardy emphasizes a paper by von Neumann, who is explicitly named as "a pupil of Hilbert," whose statement he finds "sharper and more sympathetic than Hilbert's own" (Hardy 1929, 14; see also the postscript, 25). Gandy rightly says, "because he [Hardy] wrote the text for a lecture he gives no precise references" (62). Although Hardy does not say which paper by von Neumann he has read, it seems to me pretty clear from his statement that it is a presentation of Hilbert's point of view and program, as well as the relevant dates, that it is von Neumann (1927), the very paper quoted by Gandy.

35. It is especially clear if the steps are given deterministically, as described above. But even if they are given nondeterministically, the result follows.

36. We assume some Gödel numbering of the formulae of the predicate calculus.

37. For this system, it would be simplest to replace validity by provability. But it could still be proved that any Π_1^0 sentence is equivalent to the consistency of a formula of first-order logic. Actually, and contrary to any impression given by Gandy's discussion, provability of the equivalence in the underlying system is not really needed. One could add axioms asserting the equivalence in each case. Notice also that it is not really relevant whether one can prove that the procedure always gives an answer. Gandy's other alternative that "it cannot be proved in Σ that its answer is always correct" (63) is answered in my text simply by taking this correctness as an axiom.

38. Gödel 1931, 176ff.

39. Gödel 1936.

40. The decidable case for satisfiability is statement in prenex form with just two consecutive existential quantifiers in the prefix, surrounded on both sides by strings of universal quantifiers. He goes on to show that if three consecutive existential quantifiers are allowed, a decision procedure would allow one for the predicate calculus as a whole (in modern terminology, that this is a reduction class for satisfiability). He does not state that he has shown in 1931 that such a decision procedure is impossible (though he may well have believed that such a procedure is in fact impossible, and even implicitly implied this belief).

41. However, of course Gödel could still quite consistently regard the *Entscheidungsproblem* and other such questions about effective calculability, the definition of formal system, and so on, as only decided by Turing's work. This is so even though after it had been done (after the fact, so to speak) it can be seen that the result follows from his Theorem IX.

42. At least, given Gödel's own account of his philosophical orientation.

Notice that even Gandy's late characterization of the consequences of Gödel's theorem IX is, in my opinion, still too weak.

43. Gödel refers to Turing (1936–37, 136).

44. This is the argument that, as we saw above in note 15, leads Gandy to qualify his statement of what Turing achieved in analyzing human computation. The interested reader ought to look at the entire argument, condensed here. The argument has attracted some amount of commentary, not entirely known to the present writer. See both the commentary in Gödel (1990) and by Sieg in the present volume (not known to me at the time of writing).

45. Gödel's argument seems to me to differ little in spirit from Kalmar's argument against Church's thesis. Kalmar argues (1959) in effect that every purely existential predicate (defining a recursively enumerable predicate) ought to be decidable as follows: for a positive solution, search systematically for a numerical instance for the existential quantifier. Simultaneously, for a negative solution, look for a proof by "arbitrary correct means" of the impossibility of such an instance. Is this an algorithm or a procedure? If it were, we should have used it to decide Fermat's last theorem, and should still try it on the Riemann hypothesis and the Goldbach conjecture. In fact, no doubt some people are trying this strategy, but they hardly think of it as an algorithmic procedure. Much less should one try such an "algorithm" on the halting problem or the *Entscheidungsproblem*.

46. Kleene's remarks give the impression that it is important that directions be communicated or communicable from one mathematician to another. But this seems to me to be inessential.

47. Davis writes: "We are not concerned here with attempts to distinguish 'mechanical procedures' (to which Church's thesis is held to apply) from a possible broader class of 'effective procedures" (Davis 1982, 22). Davis gives two authors as examples of what he has in mind, and, as he indicates, there are others. See also Gandy's claim, quoted above, that Turing's analysis applies to "a human being working in a routine way."

48. Moreover, Kreisel did not believe that it was a necessary condition for the acceptability of an intuitionistic system that it have the numerical instantiation property (even though standard such systems do in fact have it). For all that is required for the intuitionistic acceptability of a proof of an existential statement, even formalized in a system, is that from this proof one could obtain an intuitionistic proof of an instance, not that the proof could be formalized in the same system.

In conversation with me when he was wondering about this issue, he told me he convinced himself that for relevant intuitionistic systems, a cut elimination argument gives the "right" instance.

I have hardly exhausted the many discussions of Church's thesis, considered as a problem for the intuitionistic notion of number theoretic function, both in Kreisel's writings and in many others. The issue is simply excluded in this paper, and the discussion in the text gives one significant example.

49. What do I mean by "mathematical calculation by a machine"? I am not talking about "whether a machine can think." What I mean is that I am considering only machine programs whose successive steps follow from each other mathematically.

50. Shagrir (2002, 235) suggests two such interpretations, and attributes one of them to Sieg.

51. In fact, he gives several conditions to define his class of machines, and argues that dropping any one of them will allow the machine to exercise "free will."

52. For example, Soare (1996) and Sieg (2008).

53. I mean, for example, Newtonian physics.

54. For a quotation from Russell about this possibility see Copeland and Shagrir (2007), at the beginning of section 6. They then go on to explain how one can use what I call "Zenonian machines" to solve the halting problem.

I confess that at times past I have been sympathetic to an assumption that Zenonian calculation could be regarded as a far-fetched possibility, and that if time is finite in extent, the question of what functions are empirically computable would be meaningless.

55. When I discussed this issue in Berlin (1998; see note 1), I assumed that these were universal physical constants. I had seen this assumption in my own reading, and had already explicitly discussed the issue with an eminent physicist, including the question whether anything is known about the mathematical properties of these constants, and in particular, whether they were Turing computable. This discussion confirmed what I have to say in the text.

However, I have now heard that some physicists speculate that (say) the electron-proton mass ratio may be subject to change over time. Then what I have to say in the text would need reformulation, though I don't think it would be invalidated. I do not bother with the question further, since the point is that empirical assumptions cannot guarantee that actual machines compute Turing-computable functions, and that in fact the situation is even worse.

56. Brouwer observed, in effect, that even if we can calculate a real number with an arbitrary degree of precision, this does not mean that we can calculate all its decimal places, because a number representable with a finite decimal, not ending in 0, followed by an infinite string of 0s, can also be represented by a finite decimal with the next-lower last place, followed by a string of 9s. But on the assumptions we have just made, this possibility (and problem) is excluded (see Brouwer 1921). Note that Turing, whose original work was on computable *numbers* rather than computable functions, felt obligated in the published correction to the original paper to worry about this problem (ignored in the original paper). He refers explicitly to Brouwer (see Turing 1936–37, 153–154). I could have stated the present observations using Brouwerian approximations rather than decimals.

57. My editors have pointed out to me that similar thoughts were entertained by Kreisel and by Marian Pour-El and Ian Richards, who asked "Can one predict theoretically on the basis of some current physical theory—e.g. classical mechanics or quantum mechanics—the existence of a physical constant which is not a recursive real?" (Pour-El and Richards 1979, 63; Kreisel 1974, 11). Copeland speculated that the "magnitude of some physical quantity might conceivably be exactly τ units", where τ is a real number that is not Turing computable (Copeland 2000, 18); and computer scientist Hava Siegelmann notes that,

if there are uncomputable real constants in nature, then the "fact that the constants are not known to us or cannot even be measured is irrelevant for the true evolution of the system" (Siegelmann 2003, 110).

58. Much more is at issue than I can discuss here. But I should add that I am not myself in sympathy with the trend, prominent in some analytical philosophers, to say that mathematics ought to be justified (and can only by justified) by its applicability to physics. In fact, if we take physics at any one time, including probably today, the mathematics needed to apply to physics would include some "mathematics" that is not valid. In addition, I think that mathematics is an enterprise justified in itself, not in terms of its use in a particular other discipline. This is not the place to elaborate.

59. My attitude may be close to that of Dershowitz and Gurevich (2008).

60. All the papers quoted above that question Gandy's argument attempt to give (highly speculative) examples of empirically possible machines that go beyond Turing computability. Without knowing anything about the subject, except what I could glean from popular accounts, I have wondered whether quantum computing, an ongoing project known to be contained in Turing computing, might be an example of a hypothetical type of machine not covered by Gandy's theorem, but that might be covered by the present approach, or that of Dershowitz and Gurevich (2008). Gandy's theorem assumes a locality condition. Even though the Einstein-Podolsky-Rosen (EPR) paradox is accepted not to contradict special relativity, correlations at arbitrary distances do appear to be an essential part of quantum computing, using EPR. Thus I am not sure that Gandy's invocation of special relativity really does show that his theorem covers quantum computing. But to repeat, I am hardly really familiar with the subject.

61. Gandy worries that machines obeying Newtonian physics may violate his theorem, since he assumes special relativity, as just mentioned. But there would be no problem if each state is finitely describable in conventional first-order logic, and the operation of the machine is deterministic, so that each state follows from the previous state, and the laws governing the operation of the machine.

62. In Church (1956), section 46, Church himself lists several decidable special cases of the *Entscheidungsproblem*, largely due to others. Later, in the errata (377), Church mentions that one of his cases (case X, 257) is in fact finite (or essentially so, except for such things as changes of bound variables). Hence of course it is decidable, but Church says that there is still some interest in giving an actual decision procedure.

63. Given what we know today, there are further problems with this example, but I choose to leave the matter obscure.

64. Or rather, more exactly, that given the set of basic instructions, the conditional from any one step to the following step will follow in first-order logic.

65. However, perhaps there may be a problem in the use of such a result. After all, the original intuitive computation may itself contain unnecessary steps, that could have been, but were not in fact, eliminated. Arguably, a true translation of the argument should preserve this.

66. Though, I think I might have been able to put the matter that way. But I do not wish to obscure the simplicity of the basic point.

67. They write: "An alternative approach to proving Church's Thesis has been suggested by Kripke, based on 'Hilbert's Thesis' that 'any mathematical argument […] can be formalized in some first-order language,' and—in particular—arguments about the effects of applying the instructions of an algorithm can be so formalized" (Dershowitz and Gurevich 2008, 340). (They mean "alternative" to various approaches, such as Turing's first argument, Kolmorogov, Gandy, and others whom they cite.)

Remember that my argument about formalizability was, as much as possible, based on the statability of the steps of the argument in a first-order system, and relied on the Gödel completeness theorem for the formalizability in the stronger sense of provability in a formal system. I think I am in agreement with Dershowitz and Gurevich (2008) in this respect. But I am emphasizing deductive argument over computer models.

68. For further reading, see Gödel 1930, 1934, 2003; Kleene 1987.

69. I would like to thank the transcriber, whose identity I don't know, the editors of this volume, Jeff Buechner and Gary Ostertag for revising the original transcription, and especially Romina Padró for her help in producing the present version. This paper has been completed with support from the Saul A. Kripke Center at The Graduate Center of the City University of New York.

References

Ackermann, W. 1928. Zum Hilbertschen Aufbau der reellen Zahlen. *Mathematische Annalen* 93:118–133. English translation, with introduction, in *From Frege to Gödel: A Source Book in Mathematical Logic, 1879–1931*., ed. J. van Heijenoort, 493–507. Cambridge: Harvard University Press

Barendregt, H. 1997. The impact of the lambda calculus in logic and computer science. *Bulletin of Symbolic Logic* 3(2):181–215.

Barwise, J., ed. 1977. *Handbook of Mathematical Logic*. Amsterdam: North-Holland.

Brouwer, L. E. J. 1921. Besitzt jede reelle Zahl eine Dezimalbruchentwickelung? *Mathematische Annalen* 83:201–10. English translation in *From Brouwer to Hilbert: The Debate on the Foundations of Mathematics in the 1920s,* ed. P. Mancosu, 28–35. Oxford: Oxford University Press.

Church, A. 1936a. An unsolvable problem of elementary number theory. *American Journal of Mathematics* 58:345–63. Reprinted in Davis, *The Undecidable*, 88–107.

Church, A. 1936b. A note on the Entscheidungsproblem. *Journal of Symbolic Logic* 1(1):40–41.

Church, A. 1956. *Introduction to Mathematical Logic*. Princeton: Princeton University Press.

Copeland, B. J. 2000. Narrow versus wide mechanism. *Journal of Philosophy* 96:5–32.

Copeland, B. J., and O. Shagrir. 2007. Physical computation: How general are Gandy's principles for mechanisms? *Minds and Machines* 17:217–231.

Cutland, N. J. 1980. *Computability: An Introduction to Recursive Function Theory*. Cambridge: Cambridge University Press.

Davis, M., ed. 1965. *The Undecidable: Basic Papers on Undecidable Propositions, Unsolvable Problems and Computable Functions*. Hewlett, NY: Raven Press.

Davis, M., ed. 1982. Why Gödel didn't have Church's thesis. *Information and Control* 54:3–24.

Dershowitz, N., and Y. Gurevich. 2008. A natural axiomatization of computability and proof of Church's thesis. *Bulletin of Symbolic Logic* 14 (3):299–350.

Feferman, S. 1992. Why a Little Bit Goes a Long Way: Logical Foundations of Scientifically Applicable Mathematics. In *Proceedings of the Biennial Meeting of the Philosophy of Science Association,* Vol. 2, Symposia and Invited Papers, 442–455.

Fine, K. 1985. *Reasoning with Arbitrary Objects*. Oxford: Blackwell.

Gandy, R. 1980. Church's thesis and principles for mechanisms. In *The Kleene Symposium*, ed. J. Barwise, H. J. Keisler, and K. Kunen, 123–48. Amsterdam: North-Holland.

Gandy, R. 1988. The confluence of ideas in 1936. In *The Universal Turing Machine: A Half-Century Survey*, ed. R. Herken, 55–111. Oxford: Oxford University Press.

Gardner, M. 1970. Mathematical games: The fantastic combinations of John Conway's new solitaire game "Life." *Scientific American* 223:120–123.

Gödel, K. 1930. Die Vollständigkeit der Axiome des logischen Funktionenkalküls. *Monatshefte für Mathematik und Physik* 37:349–60. Reprinted with English translation as "The Completeness of the Axioms of the Functional Calculus of Logic" in Gödel, *Collected Works I,* 102–123.

Gödel, K. 1931. Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme, I. *Monatshefte für Mathematik und Physik* 38:173–178. Reprinted with English translation as "On formally undecidable propositions of *Principia Mathematica* and related systems I" in Gödel, *Collected Works I,* 144–95. Citations refer to the reprint.

Gödel, K. 1933. Zum Entscheidungsproblem des logischen Funktionenkalküls. *Monatshefte für Mathematik und Physik* 40:433–43. Reprinted with English translation as "On the decision problem for the functional calculus of logic" in Gödel, *Collected Works I,* 306–326.

Gödel, K. 1934. On undecidable propositions of formal mathematical systems. In Gödel, *Collected Works I,* 346–372.

Gödel, K. 1936. Über die Länge von Beweisen. *Ergebnisse eines mathematischen Kolloquiums* 7, 23–24. Reprinted with English translation as "On the length of proofs" in Gödel, *Collected Works I,* 396–98.

Gödel, K. 1972. Some remarks on the undecidability results. In Gödel, *Collected Works II,* 305–06.

Gödel, K. 1986. *Collected Works, Vol. I: Publications 1929–1936*, ed. S. Feferman, J. W. Dawson Jr., S. C. Kleene, G. H. Moore, R. M. Solovay, and J. van Heijenoort. Oxford: Oxford University Press.

Gödel, K. 1990. *Collected Works, Vol. II: Publications 1938–1974*, ed. S. Feferman, J. W. Dawson Jr., S. C. Kleene, G. H. Moore, R. M. Solovay, and J. van Heijenoort. Oxford: Oxford University Press.

Gödel, K. 1995. Undecidable Diophantine propositions. In *Collected Works, Vol. III: Unpublished Essays and Lectures*, ed. S. Feferman, J. W. Dawson Jr., W. Goldfarb, C. Parsons, and R. M. Solovay. Oxford: Oxford University Press.

Gödel, K. 2003. *Collected Works, Vol. V: Correspondence, H–Z*, ed. S. Feferman, J. W. Dawson Jr., W. Goldfarb, C. Parsons, and W. Sieg. Oxford: Clarendon Press.

Hardy, G. H. 1929. Mathematical proof. *Mind* 38(149):1–25.

Hilbert, D., and W. Ackermann. 1928. *Grundzüge der theoretischen Logik*. Berlin: Springer-Verlag. English translation, *Principles of Mathematical Logic*, 1950, ed. R. E. Luce. New York: Chelsea Publishing Company.

Kalmar, L. 1959. An argument against the plausibility of Church's thesis. In *Constructivity in Mathematics,* ed. A. Heyting, 72–80. Amsterdam: North-Holland.

Kleene, S. C. 1936. General recursive functions of natural numbers. *Mathematische Annalen* 112:727–742.

Kleene, S. C. 1952. *Introduction to Metamathematics*. Princeton: D. van Nostrand.

Kleene, S. C. 1981. The theory of recursive functions, approaching its centennial. *Bulletin of the American Mathematical Society* 5:43–61.

Kleene, S. C. 1987. Reflections on Church's thesis. *Notre Dame Journal of Formal Logic* 28:490–98.

Kleene, S. C. 1988. Turing's analysis of computability, and major applications of it. In *The Universal Turing Machine: A Half-Century Survey,* ed. R. Herken, 17–54. Oxford: Oxford University Press.

Kline, M. 1972. *Mathematical Thought from Ancient to Modern Times*. Oxford: Oxford University Press.

Kreisel, G. 1970. Church's thesis: A kind of reducibility axiom for constructive mathematics. In *Intuitionism and Proof Theory,* ed. A. Kino, J. Myhill, and R. E. Vesley, 121–50. Amsterdam: North-Holland.

Kreisel, G. 1974. A notion of mechanistic theory. *Synthese* 29:11–26.

Kripke, S. 1996. Elementary recursion theory and its applications to formal systems. Unpublished manuscript.

Martin, D. A., 1974. Sets versus classes. Unpublished manuscript.

Péter, R. 1935. Konstruktion nichtrekursiver Funktionen. *Mathematische Annalen* 111:42–60.

Post, E. L. 1944. Recursively enumerable sets of integers and their decision problems. *Bulletin of the American Mathematical Society* 50:284–316. Reprinted in Davis, *The Undecidable*, 305–337.

Pour-El, M. B., and J. I. Richards. 1979. A computable ordinary differential equation which possesses no computable solution. *Annals of Mathematical Logic* 17:61–90.

Prawitz, D. 1965. *Natural Deduction: A Proof-Theoretical Study*. Stockholm: Alqvist & Wiksell.

Robinson, R. 1948. Recursion and double recursion. *Bulletin of the American Mathematical Society* 54:987–993.

Rogers, H. 1967. *Theory of Recursive Functions and Effective Computability*. New York: McGraw-Hill.

Russell, B. 1905. On denoting. *Mind* 14:479–493.

Shagrir, O. 2002. Effective computation by humans and machines. *Minds and Machines* 12:221–240.

Shagrir, O. 2006. Gödel on Turing on computability. In *Church's Thesis after 70 Years*, ed. Olszewski, A., J. Wolenski, and R. Janusz, 393–419. Frankfurt: Ontos-Verlag.

Shoenfield, J. R. 1967. *Mathematical Logic*. Reading, MA: Addison-Wesley.

Shoenfield, J. R. 1993. *Recursion Theory*. Lecture Notes in Logic, Vol. 1. Berlin: Springer Verlag.

Sieg, W. 1994. Mechanical procedures and mathematical experience. In *Mathematics and Mind*, ed. A. George, 71–117. New York: Oxford University Press.

Sieg, W. 1997. Step by recursive step: Church's analysis of effective calculability. *Bulletin of Symbolic Logic* 3:154–180.

Sieg, W. 2008. Church without dogma: Axioms for computability. In *New Computational Paradigms: Changing Conceptions of What is Computable,* ed. S. B. Cooper, B. Löwe, and A. Sorbi, 139–152. New York: Springer.

Siegelmann, H. T. 2003. Neural and super-Turing computing. *Minds and Machines* 13:103–114.

Soare, R. 1996. Computability and recursion. *Bulletin of Symbolic Logic* 2(3):284–321.

Turing, A. M. 1936–37. On computable numbers, with an application to the *Entscheidungsproblem*. *Proceedings of the London Mathematical Society* 2(42):230–65. Reprinted in M. Davis, *The Undecidable*, 115–153.

Turing, A. M. 1950. Computing machinery and intelligence. *Mind* 59:433–460.

von Neumann, J. 1927. Zur Hilbertschen Beweistheorie. *Mathematische Zeitschrift* 26:1–46.

Wang, H. 1974. *From Mathematics to Philosophy*. London: Routledge and Kegan Paul Ltd.

Weyl, H. 1918. *Das Kontinuum.* Leipzig: Veit. English translation, *The Continuum: A Critical Examination of the Foundation of Analysis*. Trans. S. Pollard, and T. Bole. Kirksville, MO: Thomas Jefferson University Press, 1987.

Whitehead, A., and B. Russell. 1910. *Principia Mathematica*, Vol. 1. Cambridge: Cambridge University Press.

Wittgenstein, L. 1980. *Remarks on the Philosophy of Psychology*, Vol. 1. Oxford: Blackwell.