Presented at Mathematics and Mind Conference,
Amherst College, April 1991.
Unpublished.

Arguments for Church's Thesis: Comments on Wilfried Sieg

Phillip Bricker

One of the pitfalls in discussing Church's Thesis is that a number of different theses have masqueraded under that name. Church's Thesis may be formulated, following Church, as follows:

> (CT) A function of positive integers is effectively calculable if and only if it is recursive.

No problem arises from ambiguities in the term 'recursive': the thesis is purely extensional, identifying two classes; it does not matter whether the latter class of functions is picked out by recursiveness in the sense of Kleene, or Church, or Turing (to name just a few), since these all demonstrably pick out the same class. Problems do arise, however, with the interpretation of 'effectively calculable'.

Since the notion of effective calculability lies at the heart of Wilfried Sieg's discussion, I begin by examining in section 1 some of the ambiguities in this notion. In section 2, I present two standard arguments for Church's Thesis that occur in virtually every text on mathematical logic, and I attempt to point out their weaknesses. In sections 3 and 4, following Sieg's lead, I mine the classic articles of Church and Turing in search of a deeper argument. Although my interpretation of Church and Turing differs substantially from that of Sieg, I have learnt much from the study of his paper.

1. Effective calculability. First, note that 'effectively calculable' is a modal term, equivalent to 'can be effectively calculated'. Modal terms are notoriously ambiguous, and I will say something about the modality in question below. But the ambiguity I am concerned with is most naturally located elsewhere. The term 'effectively calculable' is elliptical, and doubly so. A calculation is always performed by some agent (human, mechanical, or other) and through some means. Among the possible means I include both internal capacities of the agent (e.g., vision, short-term memory) and external tools at the agent's disposal (e.g., pencil and paper, a Macintosh computer). Clearly, different ways of filling in the ellipses will result in different notions of effective calculability. The broadest notion results from quantifying existentially over possible agents and means:

> A function is effectively calculable if and only if, for some possible agent and means, there is a uniform method by which the agent can effectively calculate the function using only those means.

What is a method? Let me say at least this. A method is given by a numbered set of instructions (typically conditional and cross-referring). Call a method felicitous for some agent and means if it is within the agent's capacity, using only those means, to follow any instruction in the set. Which function if any is calculated by a method is relative to a specification of concrete inputs to represent the function's arguments, and concrete outputs to represent its values. A calculation begins when the agent, who has been provided with a set of instructions and an input, begins following the first instruction. The calculation continues step-by-step, one step for each instruction followed. The calculation ends, if at all, when the agent produces an (appropriate) output. The output represents the value of the function for the given argument. If the calculation produces no output (or an inappropriate output), then the function is undefined for the argument in question. (I use 'function of positive integers' to include both total and partial functions.)

We have still to answer the question: given a function, a method, an agent, a means, and an input-output characterization, under what conditions does the method effectively calculate that function (for that agent, means, and characterization)? I think the following five conditions would be agreed to by all. (1) The method must be felicitous (for the agent when restricted to those means).[1] The next two conditions involve two senses in which the method must be finite. (2) Calculations must be finite. For any input representing an argument for which the function is defined, the calculation produces an (appropriate) output after only finitely many steps; moreover, each such step is performed by the agent using the allowed means in a finite amount of time.[2] (3) The set of instructions must be finite. Any function of positive integers whatsoever can be calculated by an infinite "method" that includes a separate instruction (or finite set of instructions) for each input. Such a "method" is not supposed to count as effective. (4) The method must be deterministic (at least to this extent): whenever given the same input, the method produces the same result, either the same output or no output at all. If the method were not deterministic in this sense, there would not be a unique function calculated by the method. (5) The method must be correct. Given an input representing an argument for which the function is defined, the method produces an output representing the correct value of the function; given an input representing an argument for which the function is undefined, the method fails to produce an (appropriate) output.

Let us turn now to Church's Thesis. If 'effective calculability' is given its broad, non-relativistic, interpretation, CT (as formulated above) is a very strong claim. It places limits upon what can be effectively calculated by any possible agent and means. That is too strong, I think. It is open to counterexamples involving infinite beings with the power to view, as it were, all of an infinite set of positive integers at a glance (in the way that we can view, say, a seven-digit telephone number at a glance). Thus, consider any non-recursive set of positive integers, and a possible being able to view the entire set at a glance. Call such a being an oracle (for that set). The oracle can effectively calculate the (non-recursive) characteristic function of the set by following the one-step method: look (at the set) and see. Or, to vary the example, consider a possible world cohabited by human beings and oracles. In this world, a human agent (with no restriction on external means) can effectively calculate the (non-recursive) characteristic function of the set by following the one-step method: ask an oracle (for the set). Of course, nobody would take these to be counterexamples[3] to Church's Thesis as it is ordinarily intended. But that is just the point. The notion of effective calculability that people have had in mind must in some way be restricted with respect to the agent and/or the means.

Two ways of restricting the notion of effective calculability have played an especially important role in philosophical discussions of Church's Thesis. First, one might consider what is effectively calculable by a human being--or a human community, but I won't distinguish--using whatever powers of higher reasoning and intellect a human being can muster. Second, one might consider what is effectively calculable by a human being in a purely mechanical way, that is, without the use of intuition, insight, abstract reasoning, or the like.[4] Corresponding to each of these notions there is a distinct version of Church's Thesis:

(CT_H) A function of positive integers is effectively calculable by a human if and only if it is recursive.

(CT_{HM}) A function of positive integers is effectively calculable by a human in a purely mechanical way if and only if it is recursive.

Clearly, CT_H entails CT_{HM} (assuming the recursive functions can be effectively calculated by a human in a purely mechanical way). But the converse is notoriously controversial. Perhaps a function can be effectively calculated using some non-mechanical procedure (for example, involving the understanding of abstract terms on the basis of their meaning) that cannot be calculated in any purely mechanical way. Gödel thought so.[5] And others, though they may have been unconvinced by Gödel's examples, have concluded that it is an open question, as yet unanswered, perhaps unanswerable. I return to this question in section 4.

The difference between CT_H and CT_{HM} is especially relevant to the interpretation of the undecidability results of Church, Turing, and others. If CT_{HM} is true, then there is no uniform mechanical procedure for solving every mathematical problem, and so no single formal system within which all mathematical truths are provable. But unless CT_H is also true, these results do not place any limits upon what human beings might accomplish, for example, by the use of non-formal methods of proof.

How does Sieg interpret 'effectively calculable', and so Church's Thesis? Sieg never explicitly acknowledges that there are (at least) two distinct concepts between which 'effectively calculable' is ambiguous, though he sometimes uses 'effective' in a broad sense that does not entail 'mechanical', sometimes in a narrow sense that does. For example, he uses the broad sense in the introduction when he writes: "Gödel speculated how the second aspect [having to do with axioms of infinity in set theory] might give rise to a humanly effective procedure that cannot be mechanically calculated."[6] But, after presenting Turing's analysis, he claims that the"clarification of effectively calculable as calculable by a mechanical computor [i.e., human computing in a mechanical way] should be accepted." (p. 46) His reasons for the 'should' have to do with the historical context within which the concept of effective calculability developed, as presented in the first section of his paper. According to Sieg, the concept of effective calculability evolved in response to certain needs in logic and mathematics; for example, the need to characterize the concept of formal proof, and to give a precise characterization of the Entscheidungsproblem. Sieg apparently believes that these needs all converged on a single concept whose core involves "epistemological restrictions cast in 'mechanical' terms." But where Sieg sees a single unified thread, I see two distinct, though intertwining, threads. To be sure, the concept of formal proof, as developed by Frege, Gödel, and Hilbert, was essentially tied to the idea of a mechanical procedure; as Gödel wrote, "a formal system can simply be defined to be any mechanical procedure for producing formulas, called provable formulas."[7] But the Entscheidungsproblem, as traditionally posed, neither implicitly nor explicitly makes reference to mechanical procedures. For example, Sieg quotes the statement of the problem from Hilbert and Ackermann: "The Entscheidungsproblem is solved if one knows a procedure that permits the decision concerning the validity, resp. satisfiability fo a given logical expression by a finite number of operations." Only if one accepts the formalist philosophy of mathematics will one think that the solution to the Entscheidungsproblem must involve a formal or mechanical procedure.[8] I conclude that the historical context presented by Sieg supports two distinct notions of effectively calculable, and thus two distinct versions of Church's Thesis.[9]

2. Arguments for Church's Thesis. I turn to the question: what sorts of argument can be given to support Church's Thesis? Since the difference between CT_H and CT_{HM} will not be relevant for the next two sections, I will simply use 'CT' in what follows to stand equivocally for them both, and 'effectively calculable' to stand equivocally for 'effectively calculable by a human' and 'effectively calculable by a human in a mechanical way'.

It is often held that because CT relates an intuitive, imprecise notion with a notion that is formal and mathematically precise, it is not susceptible of proof. It is a definition of sorts, an explication (in Carnap's sense). That seems to be Church's view. But if by 'proof' is meant 'conclusive argument for truth', then I think the view is mistaken.

Granted, the notion of effective calculability is intuitive and informal. But if that alone made CT impossible of proof, one would have to say that the greater part of mathematics (at least, prior to the nineteenth century) contained no proofs on account of its informality and reliance on intuition. Granted, the notion of effective calculability is somewhat imprecise. The imprecision does not stem from the notion's modal character. In asking what a human can calculate, the 'can' serves to abstract from "practical" limitations involving very large (though finite) inputs or sets of instructions, and the availability of storage space. The abstractions posited, I think, are capable of precise specification. The imprecision stems rather from the imprecision of 'human' (for CT_H), and also of 'mechanical' (for CT_{HM}). But it may be that CT has the same truth value on all ways of resolving the imprecision, in which case CT could be shown to be unequivocally true or false. I know of no reason to think that the different resolutions would lead to different truth values. Granted, the notion of effective calculability is not (purely) mathematical in character, but essentially epistemological (involving human cognitive capacities). What follows from this, however, is just that CT is not susceptible of a (purely) mathematical proof; an argument need be no less conclusive for being non-mathematical. I thus see no reason to think that CT is not susceptible of proof. Indeed, it is universally accepted that one half of CT—that the recursive functions are all effectively calculable—can be conclusively established by a straightforward inductive argument, thus setting a precise "lower bound" on the class of effectively calculable functions. It would be absurd to hold that one half of CT is susceptible of proof, but not the other.[10]

Conclusive arguments for the other half of CT have been harder to come by. Traditionally, two main arguments have been given. First, there is the no counterexamples argument: mathematicians have produced many and various methods for effectively calculating functions; every function so calculated has turned out to be recursive. I suppose this gives some support to CT; but it is unclear just how much. The argument rests upon the assumption--never explicitly defended--that if there were a counterexample, it would have been discovered by now. Second, there is the argument from convergence: different and quite various attempts to provide a mathematical counterpart for effective calculability have all turned out to be extensionally equivalent, to pick out the class of recursive functions. But this argument seems to entail at most that the class of recursive functions is a mathematically natural and interesting class. The argument only supports CT on two somewhat dubious assumptions. First, it must be assumed that the intuitive notion of effective calculability is best captured by some mathematically natural class. Might it not turn out instead to be best captured by some union of natural classes, connected only by some loose family resemblance? I would argue, for example, that our intuitive notions of space and time are such. Second, even if one grants that the effectively calculable functions should turn out to form a natural class, on what grounds is it assumed that there are no other natural classes in the same neighborhood of "conceptual space," as it were, as yet undiscovered? The argument from convergence does nothing to support this.

These two arguments, then, do not seem wholly satisfactory, resting as they do on the faith that, if there were a better precise counterpart for effective calculability, it would have been discovered by now. The question thus arises: can a more conclusive argument for CT be given? Here I turn to the arguments of Church and Turing discussed by Wilfried Sieg. I begin with Church.

3. Church's Argument. Let us look in some detail at Church's discussion.[11] After proposing the identification of effective calculability and recursiveness in section 7, Church puts forward "considerations" that he thinks provide some "positive justicfication" for the identification. He introduces two notions--calculability by an algorithm and calculability within a logic[12]--and shows that, on analysis, both of these notions coincide with recursive. I will present the argument having to do with algorithms; the other argument proceeds in a parallel fashion.

Suppose we are given a (total) function F of positive integers that is calculable by an algorithm. Then, Church claims, for any input n, there is a uniquely determined sequence of expressions $E_{n1} \ldots E_{nk}$ such that: (1) each expression in the sequence is effectively calculable from the preceding expressions together with the input n; and (2) it is effectively calculable from the entire sequence and n that there is no further expression and that the output is F(n). Church then assigns Gödel numbers to expressions and sequences of expressions so that the algorithmic operations on expressions in (1) and (2) correspond to functions of positive integers. More exactly, suppose the algorithm produces, for input n, the sequence of expressions $E_{n1} \ldots E_{nk}$. Then, the following two functions encode all relevant information about the algorithm ('#' means 'Gödel number of'):

$$G(n, \#<E_{n1} \ldots E_{ni}>) = \begin{cases} \#E_{n(i+1)} & \text{if i is less than k.} \\ 10 & \text{if } i = k \\ 1 & \text{otherwise} \end{cases}$$

$$H(n, x) = \begin{cases} F(n) & \text{if } G(n, x) = 10 \\ G(n, x) & \text{otherwise} \end{cases}$$

Church then writes: "If [(P1)] the interpretation is allowed that the requirement of effective calculability which appears in our description of an algorithm means the effective calculability of the functions G and H, and if [(P2)] we take the effective calculability of G and H to mean recursiveness, then the recursiveness of F follows by a straightforward argument." The straightforward argument, I take it, could be given as follows. Given the properties of the specific Gödel numbering chosen,

$$F(n) = H(n, \text{the least } x \text{ such that } G(n,x) = 10).$$

By (P1), G and H are effectively calculable. Then, by (P2), G and H are recursive. Therefore, F is recursive because the recursive functions are closed under composition and the application of "the least x such that" operator (as had been demonstrated by Church earlier in the article).

What, if anything, has Church accomplished? Let me first consider what Sieg has to say. Suppose we preface the argument with: (P0) A function is effectively calculable if and only if it is calculable by an algorithm. This is clearly accepted by Church. Then, we seem to have a direct argument for the truth of (the "hard" half of) Church's Thesis, that all effectively calculable functions are recursive. For consider any effectively calculable function F. By (P0), F is calculable by an algorithm. By the argument explicitly presented above (using premises (P1) and (P2)), F is recursive. That is how Sieg reconstructs the argument. He combines (P1) and (P2) into what he calls Church's Central Thesis: "The steps of any effective procedure [e.g., calculation by an algorithm] must be recursive."[13] He then writes: "If this central thesis is accepted and a function is defined to be effectively calculable if, and only if, it is calculable [by an algorithm], then ... Church's ... argument <u>proves</u> that all effectively calculable functions are recursive."[14] (p. 28, Sieg's emphasis.)

I cannot accept Sieg's reconstruction of Church's argument. It would attribute to Church the most blatant <u>petitio principii</u>. The premise (P2), explicitly stated by Church, is just CT; on Sieg's reconstruction, Church simply assumes what he is trying to prove. Nor does it help to state the argument using what Sieg calls Church's central thesis. That thesis would be accepted by Church, to be sure, since it is the conjunction of the explicitly asserted (P1) and (P2). But nothing in Church's argument suggests that he had grounds for holding the "central thesis" that are independent of CT.

Sieg himself notes that, on his reconstruction, Church's argument is circular; but Sieg claims it is not viciously circular because it appeals only to a special case of what Church is trying to prove. I take it what Sieg has in mind is that the argument needs to appeal only to the following special case of CT: the effectively calculable functions that correspond (via the Gödel numbering) to steps in an algorithm are recursive.[15] I have two objections. First, Church says nothing to suggest that he intends his argument to appeal only to a special case. The individual steps in the algorithm, for all Church has said about them, may be anything at all as long as they correspond, via the Gödel numbering, to effectively calculable functions; they need not correspond to functions that are, in some sense, especially simple or elementary. When Church then assumes these effectively calculable functions are recursive, he is making use of the full generality of what he is trying to prove, not just a special case. Second, even if we suppose that Church was (implicitly) appealing to a special case of what he was trying to prove, that would not acquit the argument of vicous circularity unless the special case had been established on independent grounds. Church makes no attempt to provide such independent grounds. I conclude that Sieg's reconstruction cannot be what Church had in mind.[16]

What Church did have in mind, I think, is clear enough from his own words. He is giving reasons "for believing that [recursiveness] constitute[s] as general a characterization of [effective calculability] as is consistent with the usual intuitive understanding of it." (p. 90). Since any function that is calculable by an algorithm is, intuitively, effectively calculable, Church's proposed definition would be inconsistent with our intuitive understanding of effective calculability if some function calculable by an algorithm were not recursive. Thus, Church needs to establish that all functions calculable by an algorithm are recursive; and, indeed, that is exactly what the argument, presented above, purports to do. Church concludes the section by saying: "Thus it is shown that no more general definition of effective calculability than that proposed above [CT] can be obtained by either of two methods which naturally suggest themselves (1) by defining a function to be effectively calculable if there exists an algorithm for the calculation of its values (2) ... ." (p. 102). Since Church is arguing that CT is consistent (with certain intuitive constraints), not that it is true, there is no circularity involved in supposing CT during the course of the argument.

Has Church succeeded in giving some "positive justification" for CT? Yes, he has provided evidence that supports the "no counterexamples" argument mentioned in section 2. Since any function calculable by an algorithm is, intuitively, effectively calculable, if any such function were not recursive, it would be a counterexample to CT. Church's argument thus supports CT by showing that a natural method for attempting to generate counterexamples cannot succeed. But Church has not presented an argument that goes beyond the standard arguments discussed in section 2. That is not surprising; historically Church's paper was the principle source of those two arguments.

4. Turing's Argument. Church supports CT by arguing that certain natural attempts to generate counterexamples will not suceed. As Sieg emphasizes, Turing's argument goes much deeper.[17] According toTuring, the fundamental question is: "What are the possible processes which can be carried out in computing a number [or function]?" (p. 135). Turing imagines these processes "to be split up into 'simple operations' which are so elementary that it is not easy to imagine them further divided." (p. 136). By then providing a general characterization of such elementary operations, Turing is able to provide a (non-circular!) analysis of effective calculability (what he calls 'computability'). This allows Turing to argue, not just that no counterexamples to CT have been found, but that no counterexamples exist. It appears that Turing's analysis, if accepted, can form the basis of a more powerful argument for CT than any of the arguments considered above. Let us see why.

The problem is this: how to put a precise upper bound on the class of effectively calculable functions. One can put a lower bound on the class by providing specific examples of functions that are effectively calculable. As already noted, the recursive functions are easily shown to provide such a lower bound. But since one cannot in the same way provide examples of functions that are not effectively calculable, this method is of no help in providing an upper bound. One needs instead to characterize the class by intension, by stating conditions that are necessary for membership in the class. This characterizes from above, as it were, and so provides an upper bound on the class. If the lower bound and the upper bound can be shown to coincide, then the class of effectively calculable functions is uniquely determined.

That, in effect, is what Turing accomplished. He sets out necessary conditions, not directly on the class of effectively calculable functions itself, but on the methods for calculating its members. These necessary conditions--roughly, Principles 1.1,1.2, 2.1, 2.2, and D from Sieg's paper--serve as axioms that characterize the notion: effective procedure for calculating a function. Then, since a function is effectively calculable only if there is some effective procedure for calculating it, these axioms provide an upper bound on the class of such functions. Turing then shows that the class of functions computable by a Turing machine (i.e., of recursive functions) is such an upper bound. In other words, he shows that if a procedure satisfying conditions 1.1-2.2 and D calculates a function, then some Turing machine calculates that same function. (Sieg calls this Turing's Theorem.[18]) We now have that the recursive functions are both a lower bound and an upper bound on the effectively calculable functions, and so have established CT.

It is time to disambiguate. In the above characterization of Turing's argument, I made no mention of the agent that is carrying out the computation, or the means available to the agent. Turing is explicit that the agent in question is a human being; but he is less explicit about the means. Has Turing provided an analysis of the notion effectively calculable by a human (using whatever powers of thought), or of the notion effectively calculable by a human using purely mechanical means? In other words, is Turing's Thesis (as Sieg calls it) TT_H or TT_{HM}?

(TT_H) A human calculator satisfies the finiteness conditions 1.1-2.2.

(TT_{HM}) A human calculator using only mechanical means satisfies the finiteness conditions 1.1-2.2.

If the former, then Turing is arguing for CT_H; if the latter, for CT_{HM}. According to Sieg, Turing is asserting only TT_{HM}, and arguing only for CT_{HM}. I agree with Sieg that this is, at most, what Turing in fact establishes. But I think there is evidence that Turing intended to establish the stronger TT_H and CT_H.

Turing begins his analysis by saying "computing is normally done by writing certain symbols on paper." (p. 135). Then, the entire analysis is carried through with such a symbol manipulator in mind. Does this show that, for Turing, the calculation is to be done by purely mechanical means, without the use of abstract thought, insight, and the like? Not at all. Symbol manipulation can be done in either a mechanical or a non-mechanical way. The calculation, for Turing, is determined in part by the "state of mind" of the human calculator, and if the state of mind involves, say, the understanding of abstract concepts, then the calculation is not purely mechanical (even if it is possible to calculate the same function in a way that is purely mechanical). I thus disagree with Sieg when he says that "the context [of Turing's argument] makes crystal-clear that mechanical procedures are [being] analyzed." (p. 50). Turing nowhere explicitly requires the calculator to calculate mechanically.

Evidence that Turing did not intend to restrict his analysis to mechanical procedures is scattered throughout his article; here are two examples. The very first sentence of the article is: "The 'computable' [i.e., effectively calculable] numbers may be described briefly as the real numbers whose expressions as a decimal are calculable by finite means." (p. 116). Here and elsewhere, Turing makes clear that he is analyzing the notion of a finite procedure. But 'finite' is not equivalent in meaning to 'mechanical'. As a second example, consider how Turing concludes the section just prior to his argument: "The expression 'there is a general process for determining ...' has been used throughout this section as equivalent to 'there is a [Turing] machine which will determine'. This usage can be justified if and only if we can justify our definition of 'computable'." (p. 134). Surely, 'general process' does not mean here 'mechanical process'.

My disagreement with Sieg may be just a matter of the interpretation of Turing. Or it may be more substantial. According to Sieg, 'mechanical calculability' provides a "clarification" of 'effective calculability'. Perhaps Sieg holds that only CT_{HM} has a clear sense, not CT_H. That would be in line with his view, discussed in section 1, that the various roles in logic and mathematics that lead to the introduction of the notion of effective calculability can all be filled be a single notion, that of mechanical calculability. I have argued, on the contrary, that CT_H and CT_{HM} are distinct theses, each sufficiently clear to deserve investigation in its own right. The problem with CT_H is not that it is unclear, just unclear whether it is true. So I would reject any suggestion that CT_{HM} should replace CT_H.

My interpretation of Turing is in line with Gödel. In a well-known remark entitled "A Philosophical Error in Turing's Work,"[19] Gödel claims that Turing "gives an argument which is supposed to show that mental procedures cannot go beyond mechanical procedures." The argument in question has to do with the satisfaction of (Sieg's) condition 1.2., that there is a fixed finite number of states of mind that need to be taken into account. Gödel supports my contention that Turing is arguing here for TT_H. Since Sieg sees Turing as arguing only for TT_{HM}, he holds that Gödel's interpretation of Turing is mistaken.

I wish to look at Gödel's response in more detail, in order to assess whether he is right that Turing fails to establish TT_H, and so CT_H. The relevant passage from Turing is this: "We will also suppose that the number of states of mind which need to be taken into account [i.e., for purposes of the calculation] is finite. ... If we admitted an infinity of states of mind, some of them will be 'arbitrarily close' and will be confused." (p. 136). I find this obscure. The idea seems to be this. If the calculation requires that infinitely many (irreducibly distinct) states of mind be taken into account, then the calculator must, at any time during the calculation, be able to access them all. They must then in some way be represented in memory. But limitations on human memory make this impossible.[20] Gödel replies that Turing's argument wrongly assumes that the calculator must have access to infinitely many (irreducibly distinct) states of mind at the same time. According to Gödel, a non-mechanical process, for example, one that uses abstract terms on the basis of their meaning, might allow an (idealized) human calculator to have access (over the course of development of mathematics) to an infinite number of (irreducibly distinct) states of mind, even though at any stage in the development of mathematics, only finitely many states of mind are accessible. Such a process could allow the calculation of a non-recursive function of positive integers (presumably, by providing a decision procedure for a non-recursive set of (Gödel numbers of) true formulae). As an example, he suggests the process of "forming stronger and stronger axioms of infinity in set theory," although he concedes that this process is not yet "sufficiently understood to form a systematic, well-defined procedure." It is not my purpose here to argue for or against the epistemological plausibility of Gödel's examples of possible non-mechanical procedures. Since

Turing does nothing to rule out the possibility in question, I think Gödel is certainly justified in calling Turing's argument (for TT_H) inconclusive.

Turing presents a modification of his main argument that "avoids introducing the 'state of mind' by considering a physical and more definite counterpart of it." (p. 139). Does this modified argument escape Gödel's criticism? I think not. Gödel's criticism needs only to be modified in a parallel fashion. Turing's modified argument replaces the state of mind by "a note of instructions (written in some standard form) explaining how the work is to be continued." The idea is that, after each step of the calculation, the human calculator writes a note that will enable him, when he returns to the calculation, to complete the next step and write the next note. The "state of the system" at any stage, consisting of the sequence of symbols on the tape together with the note, is determined by the state of the system at the preceding stage. Now, suppose the calculation in question involves the sort of systematic, but non-mechanical, procedure Gödel had in mind (if any there be). Then, although there exists a (deterministic) rule governing the evolution of the system, that rule will not be formalizable; it will not be expressed within any formal system by a single axiom (or even an effectively enumerable set of axioms). Turing writes: "we assume that there is an axiom which expresses [in the functional calculus] the rules governing the behavior of the computer, in terms of the relation of the state formula at any stage to the state formula at the preceding stage. If this is so, we can construct a machine to write down the successive state formulae, and hence to compute the required number." (p. 140). But this assumption is the very point at issue. Thus, although states of mind have disappeared from the characterization of the calculation, a non-mechanical mentality will be needed in order to carry it out, in order to know how, at any stage, to write the next note. Gödel's criticism, in a modified form, still applies.

5. **Conclusion**. I have argued for the importance of distinguishing (at least) two versions of Church's Thesis, CT_H and CT_{HM}. The difference matters, for example, when considering the philosophical consequences of the undecidability results of mathematical logic. The two standard arguments for Church's Thesis presented in section 2 apply equally to either version (assuming that Gödel's speculations do not amount to a counterexample to CT_H). Since neither argument, however, is very conclusive, one is lead to seek a more powerful argument. Turing wrote: "All arguments which can be given are bound to be, fundamentally, appeals to intuition" (p. 135). That is so. But if the intuitions appealed to are clear, the argument may nonetheless be conclusive. Turing saw clearly that an argument for Church's Thesis, to be convincing, must involve an epistemological analysis of the capacities of a human calculator. I have claimed that Turing intended his analysis to apply to human calculators quite generally, and thus that he was arguing for the stronger thesis, CT_H. But even if his argument in fact only establishes CT_{HM}, that takes nothing away from his accomplishment. Turing's analysis has engendered near universal conviction that, when calculating in a purely mechanical way, only recursive functions can be calculated. And his analysis provides, in Gödel's words, "a precise and unquestionably adequate definition of the general concept of formal system." (p. 71). On the other hand, following Gödel, I have claimed that Turing's analysis gives little or no support to CT_H. An argument for CT_H might go by way of a general defense of mechanism, that the mind is, or can be simulated by, a physical machine. Mechanism, when combined with CT_M, entails CT_H. But I won't try here to assess the likelihood of finding a conclusive argument for that.

FOOTNOTES

[1]I do not require that the agent be aware of the method being used, or be able to communicate it to others; consider idiot savants, or electronic calculators. Of course, whether an agent can communicate the method, or whether we can understand the inner workings of the calculator, will bear upon the strength of our evidence that an effective method is being used.

[2]This condition is sometimes simply stated: for any input representing an argument for which the function is defined, the calculation (by the agent, using the allowed means) produces an output in a finite amount of time. But this is too weak. Consider a being who can perform infinitely many tasks in a finite amount of time, say, by performing each successive task in half the time of the preceding task. Consider the characteristic function of any non-recursive, but recursively enumerable, set of positive integers. Such a being could calculate the value of the function for any argument in a finite amount of time by listing the members of the set faster and faster, and checking whether the given argument is on the list. Such a calculation is not supposed to count as effective.

[3]Could one deny that these are counterexamples even to the strong CT on the grounds that they do not involve calculations of the function in question? Certainly, one would not ordinarily say that one calculated that 3 + 2 = 5 if one simply recalled it from memory (as we ordinarily do), or that one calculated the square root of 289 if one simply looked it up in a table. But one must be careful here; the ordinary notion cannot quite be the one at issue. For example, the successor function and identity function

are counted effectively calculable, although on many characterizations of the effectively calculable functions, their values are not calculated in the ordinary sense, but given in advance.

[4]One might also consider what is effectively calculable by a physical machine using any means permitted by the laws of physics. (Corresponding to this notion, there is a third version of Church's Thesis, CT_M.) One cannot assume without argument that what is effectively calculable by a physical machine will coincide with what is effectively calculable by a human being in a purely mechanical way. For an analysis of the former notion, see Robin Gandy, "Church's Thesis and Principles for Mechanism," in Barwise, Keisler, and Kunen, eds., The Kleene Symposium (Amsterdam: 1980) pp, 123-148.

[5]See Godel's "Postscriptum" to "On Undecidable Propositions of Formal Mathematical Systems" and his "Remarks Before the Princeton Bicentenniel Conference" in Martin Davis, ed., The Undecidable (New York: Raven Press, 1965), pp. 71-3, 84-8.

[6]Sieg cannot be using Gödel's words here. Gödel scrupulously avoided using the word 'effective'.

[7]Davis, op. cited, p. 72.

[8]Sieg quotes von Neumann, who ties the solution to the existence of a mechanical procedure; but von Neumann was then under the sway of the formalist philosophy.

[9]It should be noted, however, that modern logicians typically take 'effective' to entail 'mechanical', and thus Church's Thesis to be CT_{HM} (or CT_M). It is left to philosophers to contemplate CT_H.

[10]I here agree with Elliott Mendelson who also argues that CT may be capable of proof; but I doubt such a proof would be properly called a part of mathematics. See

"Second Thoughts About Church's Thesis and Mathematical Proofs," The Journal of Philosophy, LXXXVII, May 1990, pp. 225-33.

[11]In "An Unsolvable Problem of Elementary Number Theory," reprinted in Davis, op. cit., pp. 88-107.

[12]'Calculability within a logic' is commonly called 'representability in a theory'.

[13]This makes no sense as it stands: only functions of positive integers are recursive (for Church), not steps of an effective procedure, or algorithm. Church would have expressed the thesis something as follows: the functions that correspond, via some Godel numbering, to the steps of an algorithm are recursive.

[14]Since Sieg is discussing Church's second argument having to do with calculability in a logic, I have substituted 'calculable by an algorithm' for 'calculable in a logic'; Sieg explicitly states that what he says applies also to the case of algorithms.

[15]Note that one cannot replace 'the Godel numbering' (that is, the one Church gives in his paper) with 'some Godel numbering'. The steps in an algorithm could be made to correspond with any recursive functions whatsoever if arbitrarily complicated (yet still effective) Godel numberings are allowed.

[16]Robin Gandy appears to reconstruct Church's argument along the same lines as Sieg. See "The Confluence of Ideas in 1936," in Rolf Herken, ed., The Universal Turing Machine (Oxford: Oxford University Press, 1988), pp. 55-111.

[17]Turing's argument is in "On Computable Numbers, with an Application to the Entscheidungsproblem," reprinted in Davis, op. cit., pp. 115-151.

[18]I find it somewhat misleading to speak of "Turing's Theorem" here. It suggests that Turing's accomplishment divides into two substantial steps: the general analysis of effective procedure, and the "proof" that any effective procedure can be simulated by a

Turing machine. But this "proof" is completely trivial, as Sieg himself acknowledges. It amounts to little more than arguing that whatever can be done by an agent (or machine) that scans more than one square at a time, or moves more than one square at a time, can be done by an agent (or machine) that scans and moves one-by-one. There is really only one substantial step: the analysis of effective procedure in mechanical terms. The rest is window dressing.

[19]In Godel's Collected Works, vol. II (Oxford: Oxford University Press, 1990), p. 306.

[20]Sieg does not see Turing's claim that "the human memory is necessarily limited" as playing any role in the satisfaction of 1.2. But I do not see how else the argument could plausibly be filled out; certainly, human sensory limitations are not here at issue.