The Church-Turing Thesis

Jack Copeland

Abstract
There are various equivalent formulations of the Church-Turing thesis. A common one is that every effective computation can be carried out by a Turing machine. The Church-Turing thesis is often misunderstood, particularly in recent writing in the philosophy of mind.

Key Words: Church-Turing thesis

The Thesis and its History
The Church-Turing thesis concerns the notion of an effective or mechanical method in logic and mathematics. ‘Effective’ and its synonym ‘mechanical’ are terms of art in these disciplines: they do not carry their everyday meaning. A method, or procedure, M, for achieving some desired result is called ‘effective’ or ‘mechanical’ just in case

1. M is set out in terms of a finite number of exact instructions (each instruction being expressed by means of a finite number of symbols);
2. M will, if carried out without error, produce the desired result in a finite number of steps;
3. M can (in practice or in principle) be carried out by a human being unaided by any machinery save paper and pencil;
4. M demands no insight or ingenuity on the part of the human being carrying it out.

A well-known example of an effective method is the truth table test for tautologousness. In practice, of course, this test is unworkable for formulae containing a large number of propositional variables, but in principle one could apply it successfully to any formula of the propositional calculus, given sufficient time, tenacity, paper, and pencils.

Statements that there is an effective method for achieving such-and-such a result are commonly expressed by saying that there is an effective method for obtaining the values of such-and-such a mathematical function. For example, that there is an effective method for determining whether or not any given formula of the propositional calculus is a tautology -- e.g. the truth table method -- is expressed in function-speak by saying that there is an effective method for obtaining the values of a function, call it T, whose domain is the set of
formulae of the propositional calculus and whose value for any given formula $x$, written $T(x)$, is 1 or 0 according to whether $x$ is, or is not, a tautology.

The notion of an effective method is an informal one, and attempts to characterise effectiveness, such as the above, lack rigour, for the key requirement that the method demand no insight or ingenuity is left unexplicated. One of Turing's achievements in his paper of 1936 was to present a formally exact predicate with which the informal predicate 'can be calculated by means of an effective method' may be replaced. Church did the same (1936a). The replacement predicates that Turing and Church proposed were, on the face of it, very different from one another, but they turned out to be equivalent, in the sense that each picks out the same set of mathematical functions. The Church-Turing thesis is the assertion that this set contains every function whose values can be obtained by a method satisfying the above conditions for effectiveness. (Clearly, if there were functions of which the informal predicate, but not the formal predicate, were true, then the latter would be less general than the former and so could not reasonably be employed to replace it.) When the thesis is expressed in terms of the formal concept proposed by Turing, it is appropriate to refer to the thesis also as 'Turing's thesis'; and *mutatis mutandis* in the case of Church.

The formal concept proposed by Turing is that of *computability by Turing machine*. He argued for the claim (Turing's thesis) that whenever there is an effective method for obtaining the values of a mathematical function, the function can be computed by a Turing machine. The converse claim is easily established, for a Turing machine program is itself a specification of an effective method: without exercising any ingenuity or insight, a human being can work through the instructions in the program and carry out the required operations. If Turing's thesis is correct, then talk about the existence and non-existence of effective methods can be replaced throughout mathematics and logic by talk about the existence or non-existence of Turing machine programs.

Turing stated his thesis in numerous places, with varying degrees of rigour. The following formulation is one of the most accessible.

**Turing's thesis:**

*LCMs [logical computing machines: Turing's expression for Turing machines] can do anything that could be described as "rule of thumb" or "purely mechanical".* (Turing 1948:7.)

He adds:

*This is sufficiently well established that it is now agreed amongst logicians that "calculable by means of an LCM" is the correct accurate rendering of such phrases. (1948: 7.)*

Turing introduced this thesis in the course of arguing that the Entscheidungsproblem, or decision problem, for the predicate calculus - posed by Hilbert (Hilbert and Ackermann 1928) -- is unsolvable. Here is Church's account of the Entscheidungsproblem:

*By the Entscheidungsproblem of a system of symbolic logic is here understood the problem to find an effective method by which, given any expression $Q$ in the notation of the system, it can be determined whether or not $Q$ is provable in the system. (Church 1936b: 41.)*

The truth table test is such a method for the propositional calculus. Turing showed that, given his thesis, there can be no such method for the predicate calculus. He proved formally that there is no Turing machine which can determine, in a finite number of steps,
whether or not any given formula of the predicate calculus is a theorem of the calculus. So, given his thesis that if an effective method exists then it can be carried out by one of his machines, it follows that there is no such method to be found.

Church had arrived at the same negative result a few months earlier, employing the concept of lambda-definability in place of computability by Turing machine. (A function of positive integers is said to be lambda-definable if the values of the function can be calculated by a process of repeated substitution.) Church and Turing discovered the result quite independently of one another. Turing's method of obtaining it is rather more satisfying than Church's, as Church himself acknowledged in a review of Turing's work:

"computability by a Turing machine ... has the advantage of making the identification with effectiveness in the ordinary (not explicitly defined) sense evident immediately." (1937a: 43.)

(Another aspect in which their approaches differ is that Turing's concerns were rather more general than Church's, in that the latter considered only functions of positive integers (see below), whereas Turing described his work as encompassing "computable functions of an integral variable or a real or computable variable, computable predicates, and so forth" (1936: 230). He intended to pursue the theory of computable functions of a real variable in a subsequent paper, but in fact did not do so.)

Church used the (informal) expression 'effectively calculable' to indicate that there is an effective method for calculating the values of the function. He proposed that we define the notion ... of an effectively calculable function of positive integers by identifying it with the notion of a recursive function of positive integers (or of a lambda-definable function of positive integers). (1936a: 356.)

The concept of a lambda-definable function is due to Church and Kleene (Church 1932, 1936a, 1941, Kleene 1935) and the concept of a recursive function to Gödel and Herbrand (Gödel 1934, Herbrand 1932). The class of lambda-definable functions and the class of recursive functions are identical. This was established in the case of functions of positive integers by Church and Kleene (Church 1936a, Kleene 1936). After learning of Church's proposal, Turing quickly established that the apparatus of lambda-definability and his own apparatus of computability are equivalent (1936: 263ff). Thus, in Church's proposal, the words 'recursive function of positive integers' can be replaced by the words 'function of positive integers computable by Turing machine'.

Post referred to Church's identification of effective calculability with recursiveness as a "working hypothesis", and quite properly criticised Church for masking this hypothesis as a definition.

"To mask this identification under a definition ... blinds us to the need of its continual verification." (Post 1936: 105.)

This, then, is the "working hypothesis" that, in effect, Church proposed:

**Church's thesis:**

A function of positive integers is effectively calculable only if recursive.

The reverse implication, that every recursive function of positive integers is effectively calculable, is commonly referred to as the converse of Church's thesis (although Church himself did not so distinguish, bundling both theses together in his 'definition'). If attention is restricted to functions of positive integers then Church's thesis and Turing's
thesis are equivalent, in view of the previously mentioned results by Church, Kleene and Turing.

The term ‘Church-Turing thesis’ seems to have been first introduced by Kleene, with a small flourish of bias in favour of Church:

*So Turing’s and Church’s theses are equivalent. We shall usually refer to them both as Church’s thesis, or in connection with that one of its ... versions which deals with ‘Turing machines’ as the Church-Turing thesis. (Kleene 1967: 232.)*

Much evidence has been amassed for the ‘working hypothesis’ proposed by Church and Turing in 1936. One of the fullest surveys is to be found in chapters 12 and 13 of Kleene (1952). In summary: (1) Every effectively calculable function that has been investigated in this respect has turned out to be computable by Turing machine. (2) All known methods or operations for obtaining new effectively calculable functions from given effectively calculable functions are paralleled by methods for constructing new Turing machines from given Turing machines. (3) All attempts to give an exact analysis of the intuitive notion of an effectively calculable function have turned out to be equivalent in the sense that each analysis offered has been proved to pick out the same class of functions, namely those that are computable by Turing machine. Because of the diversity of the various analyses the latter is generally considered strong evidence. For example, apart from the analyses already mentioned in terms of lambda-definability and recursiveness, there are analyses in terms of register machines (Shepherdson and Sturgis 1963), Post’s canonical and normal systems (Post 1943, 1946), combinatory definability (Schönfinkel 1924, Curry 1929, 1930, 1932), Markov algorithms (Markov 1960), and Gödel’s notion of reckonability (Gödel 1936, Kleene 1952).

While there have from time to time been attempts to call the Church-Turing thesis into question (for example by Kalmar (1959); Mendelson (1963) replies), the summary of the situation that Turing gave in 1948 is no less true today: "it is now agreed amongst logicians that ‘calculable by means of an LCM’ is the correct accurate rendering" (of the informal notion in question).

**Misunderstandings of the Thesis**

A myth seems to have arisen concerning Turing’s paper of 1936, namely that he there gave a treatment of the limits of mechanism and established a fundamental result to the effect that the universal Turing machine can simulate the behaviour of any machine. The myth has passed into the philosophy of mind, generally to pernicious effect. For example, the *Oxford Companion to the Mind* states: "Turing showed that his very simple machine ... can specify the steps required for the solution of any problem that can be solved by instructions, explicitly stated rules, or procedures" (Gregory 1987: 784). Dennett maintains that "Turing had proven - and this is probably his greatest contribution - that his Universal Turing machine can compute any function that any computer, with any architecture, can compute" (1991: 215); also that every "task for which there is a clear recipe composed of simple steps can be performed by a very simple computer, a universal Turing machine, the universal recipe-follower (1978: xviii). Paul and Patricia Churchland assert that Turing’s "results entail something remarkable, namely that a standard digital computer, given only the right program, a large enough memory and sufficient time, can compute any rule-governed input-output function. That is, it can display any systematic pattern of responses to the
environment whatsoever” (1990: 26). These various quotations are typical of current writing on the foundations of the computational theory of mind. It seems on the surface unlikely that these authors mean to restrict the general notions of ‘explicitly stated rule’, ‘instruction’, ‘clear recipe composed of simple steps’, ‘computer with any architecture’, ‘rule-governed function’ and ‘systematic pattern’ so as to apply only to things that can be obeyed, simulated, calculated, or produced by a machine that implements ‘effective’ methods in Turing’s original sense. But unless these notions are restricted in this way from the start, we should reject such claims.

Turing did not show that his machines can solve any problem that can be solved “by instructions, explicitly stated rules, or procedures”, nor did he prove that the universal Turing machine "can compute any function that any computer, with any architecture, can compute". He proved that his universal machine can compute any function that any Turing machine can compute; and he put forward, and advanced philosophical arguments in support of, the thesis here called Turing’s thesis. But a thesis concerning the extent of effective methods -- which is to say, concerning the extent of procedures of a certain sort that a human being unaided by machinery is capable of carrying out -- carries no implication concerning the extent of the procedures that machines are capable of carrying out, even machines acting in accordance with ‘explicitly stated rules’. For among a machine’s repertoire of atomic operations there may be those that no human being unaided by machinery can perform.

The further proposition, very different from Turing’s own thesis, that a Turing machine can compute whatever can be computed by any machine working on finite data in accordance with a finite program of instructions, is sometimes also referred to as (a version of) the Church-Turing thesis or Church’s thesis. For example, Smolensky says:

connectionist models ... may possibly even challenge the strong construal of Church’s Thesis as the claim that the class of well-defined computations is exhausted by those of Turing machines. (Smolensky 1988: 3.)

This loosening of established terminology is unfortunate, for neither Church nor Turing endorsed, or even formulated, this further proposition. There are numerous examples of this extended usage in the literature. The following are typical.

That there exists a most general formulation of machine and that it leads to a unique set of input-output functions has come to be called Church’s thesis. (Newell 1980: 150.)

"The work of Church and Turing fundamentally connects computers and Turing machines. The limits of Turing machines, according to the Church-Turing thesis, also describe the theoretical limits of all computers. (McArthur 1991: 401.)

"[I]t is difficult to see how any language that could actually be run on a physical computer could do more than Fortran can do. The idea that there is no such language is called Church’s thesis. (Geroch and Hartle 1986: 539.)"

Also (more distant still from anything that Church or Turing actually wrote):

I can now state the physical version of the Church-Turing principle: “Every finitely realizable physical system can be perfectly simulated by a universal model computing machine operating by finite means.” This formulation is both better defined and more physical than Turing’s own way of expressing it. (Deutsch 1985: 99.)
This formulation may be 'more physical' than Turing's own, but it is scarcely 'better defined'. The notion of an effective method played an important role in early debates about the foundations of mathematics, and it was sufficiently clear to allow Turing, Church, and others to recognize that different formal accounts gave alternative modellings of the notion. Their notion was certainly not that of a 'finitely realizable physical system'.

Gandy (1980) is one of the few writers to distinguish explicitly between Turing's thesis and the stronger proposition that whatever can be calculated by a machine can be calculated by a Turing machine. Borrowing Gandy's terminology, I will call the stronger proposition 'Thesis M'. I will use expressions such as 'the Church-Turing thesis properly so-called' for the proposition that Church and Turing themselves endorsed.

**Thesis M:**
Whatever can be calculated by a machine (working on finite data in accordance with a finite program of instructions) is Turing-machine-computable.

Thesis M itself admits of two interpretations, according to whether the phrase "can be generated by a machine" is taken in the narrow, this-worldly, sense of "can be generated by a machine that conforms to the physical laws (if not to the resource constraints) of the actual world", or in a wide sense that abstracts from the issue of whether or not the notional machine in question could exist in the actual world. Under the latter interpretation, thesis M is false. It is straightforward to describe notional machines, or 'hypercomputers' (Copeland and Proudfoot (1999a)) that generate functions not Turing-machine-computable (see e.g. Abramson (1971), Copeland (2000), Copeland and Proudfoot (2000), Stewart (1991)). It is an open empirical question whether or not the narrow this-worldly version of thesis M is true. Speculation that there may be physical processes -- and so, potentially, machine-operations -- whose behaviour conforms to functions not computable by Turing machine stretches back over at least five decades; see, for example, da Costa and Doria (1991), (1994), Doyle (1982), Geroch and Hartle (1986), Hogarth (1994), Kreisel (1967), (1974), (1982), Pour-El and Richards (1979), (1981), Scarpellini (1963), Siegelmann and Sontag (1994), and Stannett (1990). (Copeland and Sylvan (1999) is a survey; see also Copeland and Proudfoot (1999b).)

The literature on the computational theory of the mind contains numerous endorsements of propositions equivalent or similar to thesis M that are supported by nothing more than a nod toward Turing or Church (as is illustrated by a number of the quotations given earlier). Perhaps some writers are simply misled by the terminological practice whereby a thesis concerning which there is little real doubt, the Church-Turing thesis properly so called, and a different thesis of unknown truth-value, are referred to indiscriminately as Church's thesis or the Church-Turing thesis -- albeit with accompanying hedges like 'strong form' and 'physical version'. Other writers may maintain thesis M (or some equivalent or near equivalent) on the spurious grounds that the various, and prima facie very different, attempts -- by Turing, Church, Post, Markov, and others -- to characterise in precise terms the informal notion of an effective procedure have turned out to be equivalent to one another. This is evidence concerning the extent of effective procedures, and not evidence concerning the extent of what can be calculated by machine.
The error of confusing the Church-Turing thesis properly so-called with thesis M has led to some remarkable claims in the foundations of psychology. For example, one frequently encounters the view that psychology must be capable of being expressed ultimately in terms of the Turing machine (e.g. Fodor 1981: 130; Boden 1988: 259). To one who makes the error, conceptual space will seem to contain no room for mechanical models of the mind that are not equivalent to Turing machines. Yet it is certainly possible that psychology will find the need to employ models of human cognition that transcend Turing machines.

Note that in some cases, an author’s apparent endorsement of M is merely apparent. In this connection, it is important to remember that in the technical literature the word ‘computable’ is often tied by definition to effective calculability. Thus a function is said to be computable if and only if there is an effective procedure for determining its values. Accordingly, a common formulation of the Church-Turing thesis in the technical literature and in textbooks is:

All computable functions are computable by Turing machine.

Corollaries such as the following are sometimes offered:

certain functions are uncomputable in an absolute sense: uncomputable even by [Turing machine], and, therefore, uncomputable by any past, present, or future real machine. (Boolos and Jeffrey 1980: 55.)

Given the definition of ‘computable’ as ‘effectively calculable’, the Church-Turing thesis does entail that if a function f is not computable by Turing machine then it is not computable by any machine. However, to a casual reader of the technical literature, such statements may appear to say more than they in fact do. (Of course, the decision to tie the term ‘computable’ and its cognates to the concept of effectiveness does not settle the truth-value of thesis M. Those who abide by this terminological decision are simply prevented from describing a machine that falsifies thesis M as computing the function that it generates.)

The word ‘mechanical’, too, in technical usage, is tied to effectiveness and, as already remarked, ‘mechanical’ and ‘effective’ are used interchangeably. (Gandy (1988) outlines the history of this usage of the word ‘mechanical’.) Thus statements like the following are to be found in the technical literature:

Turing proposed that a certain class of abstract machines could perform any ‘mechanical’ computing procedure. (Mendelson 1964: 229.)

Understood correctly, this remark attributes to Turing not thesis M but the Church-Turing thesis. This usage of ‘mechanical’ tends to obscure the possibility that there may be machines, or biological organs, that calculate (or compute, in a broad sense) functions that are not Turing-machine-computable. For the question ‘Can a machine execute a procedure that is not mechanical?’ may appear self-answering, yet this is precisely what is asked if thesis M is questioned.

Thesis M is not the only problematic thesis that is linked to the Church-Turing thesis. An error which, unfortunately, is common in modern writing on computability and the brain is to hold that Turing’s results somehow entail that the brain, and indeed any biological or physical system whatever, can be simulated by a Turing machine. For example, the entry on Turing in the recent A Companion to the Philosophy of Mind contains the following claims: “we can depend on there being a Turing machine that captures the
functional relations of the brain", for so long as "these relations between input and output are functionally well-behaved enough to be describable by ... mathematical relationships ... we know that some specific version of a Turing machine will be able to mimic them" (Guttenplan 1994: 595). Searle writes in a similar fashion:

Can the operations of the brain be simulated on a digital computer? ... The answer seems to me ... demonstrably 'Yes' ... That is, naturally interpreted, the question means: Is there some description of the brain such that under that description you could do a computational simulation of the operations of the brain. But given Church's thesis that anything that can be given a precise enough characterization as a set of steps can be simulated on a digital computer, it follows trivially that the question has an affirmative answer. (Searle 1992: 200.)

So too Johnson-Laird, and the Churchlands:

If you assume that [consciousness] is scientifically explicable ... [and] granted that the [Church-Turing] thesis is correct, then ... [i]f you believe [functionalism] to be false ... then ... you [should] hold that consciousness could be modelled in a computer program in the same way that, say, the weather can be modelled ... [and if] you accept functionalism ... you should believe that consciousness is a computational process. (Johnson-Laird 1987: 252.)

Church's Thesis says that whatever is computable is Turing computable. Assuming, with some safety, that what the mind-brain does is computable, then it can in principle be simulated by a computer. (Churchland and Churchland 1983: 6.)

As previously mentioned, Churchland and Churchland seem to believe, erroneously, that Turing's "results entail ... that a standard digital computer, given only the right program, a large enough memory and sufficient time, can ... display any systematic pattern of responses to the environment whatsoever" (1990: 26). (They do not explicitly restrict talk of 'systematic patterns' to ones that are effectively calculable.) This no doubt explains why they think they can assume "with some safety" that what the mind-brain does is computable, for on their understanding of matters this is to assume only that the mind-brain exhibits a systematic pattern of responses, or is characterised by a 'rule-governed' (1990: 26) input-output function.

The Church-Turing thesis does not entail that the brain (or the mind, or consciousness) can be modelled by a Turing machine program, not even in conjunction with the belief that the brain (or mind, etc.) is scientifically explicable, or exhibits a systematic pattern of responses to the environment, or is 'rule-governed' (etc.). Each of the authors quoted seems to be assuming the truth of a close cousin of thesis M, which I will call

**Thesis S**: Any process that can be given a mathematical description (or that is scientifically describable or scientifically explicable) can be simulated by a Turing machine.

As with thesis M, neither the Church-Turing thesis properly so-called nor any result proved by Turing or Church entails thesis S. This is so even when the thesis is taken narrowly, as concerning processes that conform to the physics of the real world. (Thesis S taken in the wide sense is known to be false; see the references given earlier re the wide version of thesis M.) Any device or organ whose internal processes can be described completely by means of effectively calculable functions can be simulated exactly by a Turing machine program (provided that the input into the device or organ is itself
Turing-machine-computable, which is to say, is either finite or expressible as a computable number, in Turing's sense (which is explained below); but any device or organ whose mathematical description involves functions that are not effectively calculable cannot be so simulated. As Turing showed, there are uncountably many such functions. (Examples from logic are Turing's famous halting function (described in the entry on Turing machines) and the function $D$ whose domain is the set of well-formed formulae of the predicate calculus and whose values, $D(x)$, are 1 or 0 according to whether $x$ is, or is not, derivable from the Bernays-Hilbert-Ackermann axioms for predicate logic.) It is an open question whether a completed neuroscience will employ functions that are not effectively calculable.

Some Key Remarks by Turing

Turing introduces his machines with the intention of providing an idealised description of a certain human activity, the tedious one of numerical computation, which until the advent of automatic computing machines was the occupation of many thousands of people in business, government, and research establishments. He prefaces his first description of a Turing machine with the words:

*We may compare a man in the process of computing a ... number to a machine.*

(Turing 1936: 231.)

The Turing machine is a model, idealised in certain respects, of a human being calculating in accordance with an effective procedure. Wittgenstein put this point in a striking way:

*Turing's "Machines": These machines are humans who calculate.* (Wittgenstein 1980, 1096.)

It is a point that Turing was to emphasise, in various forms, again and again. For example:

*A man provided with paper, pencil, and rubber, and subject to strict discipline, is in effect a universal machine.* (Turing 1948: 9.)

The electronic stored-program digital computers for which the universal Turing machine was a blueprint are, each of them, computationally equivalent to a Turing machine, and so they too are, in a sense, models of human beings engaged in computation. Turing chose to emphasise this when explaining these electronic machines in a manner suitable for an audience of uninitiates:

*The idea behind digital computers may be explained by saying that these machines are intended to carry out any operations which could be done by a human computer.* (Turing 1950a: 436.)

He makes the point a little more precisely in the technical document containing his preliminary design for the Automatic Computing Engine or ACE. (The ACE was an electronic stored-program computer built at the National Physical Laboratory, London. A pilot version first ran in 1950 and at the time was the fastest computer in the world. The commercial model was called the DEUCE.)

*The class of problems capable of solution by the machine [the ACE] can be defined fairly specifically. They are [a subset of] those problems which can be solved by human clerical labour, working to fixed rules, and without understanding.* (Turing 1946: 38-9.)
available to the human clerk.) It was presumably because he considered the point under discussion to be essential for understanding the nature of the new electronic machines that he chose to begin his Programmers’ Handbook for Manchester Electronic Computer with this explanation:

Electronic computers are intended to carry out any definite rule of thumb process which could have been done by a human operator working in a disciplined but unintelligent manner. (Turing 1950b: 1.)

It was not some deficiency of imagination that led Turing to model his computing machines on what could be achieved by a human computer. The purpose for which the Turing machine was invented demanded it. The Entscheidungsproblem is the problem of finding a *humanly executable* procedure of a certain sort, and Turing’s aim was precisely to show that there is no such procedure in the case of predicate logic. He proved that no Turing machine can compute the values of the function D that I described earlier, and he argued that his model of human computation is sufficiently general, in the sense that there are no intuitively computable (i.e. effectively calculable) functions that Turing machines are incapable of computing.

The latter claim is, of course, Turing’s thesis. Here are two additional formulations of the thesis, from his paper of 1936.

[T]he “computable numbers” [the numbers whose decimal representations can be generated progressively by a Turing machine] include all numbers which would naturally be regarded as computable. (Turing 1936: 249.)

It is my contention that these operations [the primitive operations of a Turing machine] include all those which are used in the computation of a number: (Turing 1936: 232.)

(As Turing explains: "Although the subject of this paper is ostensibly the computable numbers, it is almost equally easy to define and investigate computable functions ... I have chosen the computable numbers for explicit treatment as involving the least cumbersome technique" (1936: 230).)

To understand these assertions as Turing intended them it is essential to keep in mind that when he uses the words ‘computer’, ‘computable’ and ‘computation’ he employs them not in their modern sense as pertaining to machines but as pertaining to human calculators. Many passages make this obvious.

Computers always spend just as long in writing numbers down and deciding what to do next as they do in actual multiplications, and it is just the same with ACE ... [T]he ACE will do the work of about 10,000 computers ... Computers will still be employed on small calculations ... (Turing 1947: 116, 120.)

Thus when Turing maintains that every number or function that "would naturally be regarded as computable" can be calculated by a Turing machine he is asserting not thesis M but a thesis concerning the extent of the effectively calculable numbers and functions. Similarly, when Church writes (in a review of Post (1936)):

To define effectiveness as computability by an arbitrary machine, subject to restrictions of finiteness, would seem to be an adequate representation of the ordinary notion (Church 1937b: 43),

he is to be understood not as entertaining some form of thesis M but as endorsing
the identification of the effectively calculable functions with those functions that can be calculated by an arbitrary machine whose principles of operation are such as to mimic the actions of a human computer. (There is much that is ‘arbitrary’ about the machines described (independently, in the same year) by Turing and Post, for example the one-dimensional arrangement of the squares of the tape (or in Post’s case, of the ‘boxes’), the absence of a system of addresses for squares of the tape, the choice between a two-way and a one-way infinite tape, and, in Post’s case, the restriction that a square admit of only two possible conditions, blank or marked by a single vertical stroke.)

It is equally important to note also that when Turing uses the word ‘machine’ he often means not machine-in-general but, as we would now say, Turing machine. At one point he explicitly draws attention to this idiosyncratic usage:

*The expression "machine process" of course means one which could be carried out by the type of machine I was considering [in Turing (1936)]. (Turing 1947: 107.)*

Thus when, a few pages later, he asserts that "machine processes and rule of thumb processes are synonymous" (1947: 112), he is to be understood as advancing the Church-Turing thesis (and its converse), not a version of thesis M. Unless his intended usage is borne in mind, misunderstanding is certain to ensue. Especially liable to mislead are statements like the following, which a casual reader might easily mistake for a formulation of thesis M:

*The importance of the universal machine is clear. We do not need to have an infinity of different machines doing different jobs. A single one will suffice. The engineering problem of producing various machines for various jobs is replaced by the office work of "programming" the universal machine to do these jobs. (Turing 1948: 7.)*

In context it is perfectly clear that these remarks concern machines equivalent to Turing machines (the passage is embedded in a discussion of LCMs).

Whether or not Turing would, if queried, have assented to thesis M is unknown. There is certainly no textual evidence in favour of the common belief that he did so assent.
REFERENCES

Abramson FG. Effective Computation over the Real Numbers. Twelfth Annual Symposium on Switching and Automata Theory. Northridge, Calif.: Institute of Electrical and Electronics Engineers. 1971.


Schönfinkel M. Über die Bausteine der mathematischen. Mathematische Annalen 1924;92:305-316.


Turing AM. Computing Machinery and Intelligence. Mind 1950a;59:433-460.


