J Gen Philos Sci (2014) 45:239–262
DOI 10.1007/s10838-014-9244-x

ARTICLE

# Computable Diagonalizations and Turing's Cardinality Paradox

**Dale Jacquette**

Published online: 26 February 2014


**Abstract** A. N. Turing's 1936 concept of computability, computing machines, and computable binary digital sequences, is subject to *Turing's Cardinality Paradox*. The paradox conjoins two opposed but comparably powerful lines of argument, supporting the propositions that the cardinality of dedicated Turing machines outputting all and only the computable binary digital sequences can only be denumerable, and yet must also be nondenumerable. Turing's objections to a similar kind of diagonalization are answered, and the implications of the paradox for the concept of a Turing machine, computability, computable sequences, and Turing's effort to prove the unsolvability of the *Entscheidungsproblem*, are explained in light of the paradox. A solution to Turing's Cardinality Paradox is proposed, positing a higher geometrical dimensionality of machine symbol-editing information processing and storage media than is available to canonical Turing machine tapes. The suggestion is to add volume to Turing's discrete two-dimensional machine tape squares, considering them instead as similarly ideally connected massive three-dimensional machine information cells. Three-dimensional computing machine symbol-editing information processing cells, as opposed to Turing's two-dimensional machine tape squares, can take advantage of a denumerably infinite potential for parallel digital sequence computing, by which to accommodate denumerably infinitely many computable diagonalizations. A three-dimensional model of machine information storage and processing cells is recommended on independent grounds as better representing the biological realities of digital information processing isomorphisms in the three-dimensional neural networks of living computers.

**Keywords** Cantor, Georg · Cardinality · Computability · Computable sequence · Denumerability, nondenumerability · Diagonalization · *Entscheidungsproblem* · Löwenheim-Skolem theorem · Skolem paradox · Turing, A. N. · Universal Turing machine

D. Jacquette (✉)
University of Bern, Bern, Switzerland
e-mail: dale.jacquette@philo.unibe.ch

## 1 Ideal Symbol-Editing Machines

Turing's (1936) classic essay, 'On Computable Numbers, With an Application to the Entscheidungsproblem', is a landmark contribution to mathematical logic and logical metatheory. Turing's discussion is prominent historically among a small number of formal studies in mathematical machine and computation theory that inaugurated the present age of information sciences. It is the foundation stone for the concept of a computable real number, and of all that can be done with algorithmic symbol-editing of computable real number sequences. Which is to say, whatever can be done with the most powerful digital information processing machines.[1]

Among other important innovations, Turing is the first to explain how an ideal symbol-editing device, capable of only a few simple mechanical operations, can generate all of the computable binary digit symbol string sequences belonging to all and only the computable real numbers. Turing machines dedicated (DTMs) to computing specific real number sequences (CSs) accomplish their respective ideal mechanical computational tasks autonomously, without intelligent intervention. They produce symbol strings independent of intelligent observation or control, by means of implementing finitely many precisely specifiable step-by-step elementary algorithmic machine instructions in infinitely many cycles. It is such, for simplicity throughout, infinitely, otherwise, indefinitely or potentially infinitely, cycling finite program instructions for a DTM that under further necessary conditions produces a complete CS only as the DTM's output printing of binary digits approaches infinity.

We shall continue to speak of DTMs in this connection as ideal rather than abstract. Although, like abstract entities, DTMs are not themselves supposed to be physical objects; unlike abstract entities, they are also not conceptually timeless and spaceless. They involve step-by-step sequential algorithmic operations that extend over considered ideal rather than real physical time and space, in which one condition is first satisfied and then another part of the DTM's program is implemented by ideal printing onto an ideal machine tape of infinitely many squares, and so on. The machine's symbol processing continues infinitely sequentially in precisely those applications where a DTM outputs a CS. The individual CSs produced by circle-free DTMs, on the other hand, can equally be described as ideal or abstract. Unlike the circle-free DTMs in the process of making CSs, CSs themselves are symbol strings on ideal machine tapes representing real number digital sequences that, like other numbers in the default Platonic ontology, can be regarded as abstract entities transcending physical time and space. The CSs, for one thing, must be infinite in extent, and we should need a physical machine tape to be simultaneously renovated and restored against entropy at infinitely many sites. The point in any case is not to have a DTM *create* a CS. We are free to suppose that every CS as a computable real number sequence exists independently of its DTMs in a Platonic order of abstract entities. The idea is to show by the existence of a DTM that we can formally single out exactly these digital sequences as computable, distinguished from all other remaining digital sequences as random. This rigorously developed concept alone marks Turing's essay as a historic contribution to the logical foundations of mathematics.[2]

[1] Turing (1936).

[2] On Turing's contributions to computing theory from a cultural perspective, see Hodges (1983). Contemporaries to have arrived at similar if less developed versions of Turing's concept historically and almost simultaneously include Church (1936a, b) and Post (1936).

DTMs that continue computing to infinity, endlessly printing digits 0 or 1 in a denumerably infinite symbol string, a computable sequence or CS, on their infinitely extended machine tapes, according to finitely describable algorithmic machine instructions, Turing designates as 'circle-free', while all other DTMs are said to be 'circular'. Circular DTMs either halt at some stage of carrying out their machine instructions, or enter into symbol-editing processes in which the algorithmic printing of a denumerably infinite series of 0s and 1s never occurs, or after a finite run is somehow discontinued, and in either case fails to produce a CS. This can happen in a variety of ways, although the reason is always the DTM's faulty program instructions. A circular machine can either stop altogether, or cycle endlessly without producing any or more than a finite number of rule-governed printings of 0s or 1s. Otherwise, a circular DTM can continue endlessly to edit and print and/or erase other, variable, symbols, from the finite vocabulary of symbols in the DTM's machine language, in programmed operations that never result in a CS.[3]

## 2 Turing Machine Architecture

Turing machines are often imaginatively pictured as functioning by means of an information processing tape 'head' that is capable of 'reading' or 'scanning' a symbol recorded on a tape. The head responds mechanically in definitely prescribed ways to the physical configuration of certain further specific symbol inscriptions the head encounters as it moves along from tape square to tape square according to prescribed parts of its machine instructions, 'writing' or 'printing' and 'erasing' certain symbols, on or from the machine tape as it reads and edits symbols encountered in its movements along the tape. The machine tape consists of no more than denumerably infinitely many 'squares' or discrete information storage and processing spaces, arranged in a definite linear order, to which the machine's instructions are automatically applied.

Thereafter, upon completing a scanning or symbol-editing instruction, the machine head moves along or directs the tape to move along to another discrete information processing square in one of two opposite linear directions. There, depending on its programmed instructions, the machine is directed to either halt or continue its operations in theory denumerably infinitely many times, cycling through denumerably infinitely many executions of its machine instructions on the continuously algorithmically modified symbol content of its machine tape that grows to infinity as the CS of a computable real number is calculated by the right sequences of symbol-editing instructions. A DTM head can read a symbol in a discrete square on a tape of denumerably infinitely many potential symbol spaces, and thereupon alternatively leave a scanned symbol in place or erase it, if erasure is permitted for a symbol appearing in the processing and computing squares rather than the CS-building squares. Binary digits or what Turing also calls symbols of 'the first kind', printed in the so-called *F*-squares on the DTM machine tape are off-limits to erasure as the advancing digits of the CS are printed on alternating *F*-squares of the DTM's machine tape. Whereas auxiliary variables and other symbols other than binary digits, which Turing classifies as symbols of 'the second kind', together with binary digits or symbols of the first kind, temporarily inscribed in the calculation scratchpad spaces or *E*-squares on the machine tape to facilitate the execution of symbol-editing instructions, can always be both printed and erased as the DTM's specific algorithmic machine instructions require.[4] The

[3] Turing (1936, 233).

[4] Ibid., 232–235.

DTM head can also print another symbol in an *E*-square, from any available to it in its output palette, before moving along or having the tape itself move from left to right, or right to left, one square at a time. The DTM's finite algorithmic machine instructions presented in a finitely surveyable table, are incrementally executed, and, as Turing insists, the DTM functions automatically, as though in ideal time, with no external intelligent or further mechanical intervention. A circle-free DTM can proceed infinitely toward the step-by-step construction of a CS for a computable real number like π only if it loops infinitely many times through parts of its finite machine instruction table. The fact that such a DTM is circle-free signifies that it is constantly presenting itself with evolving output to take as new input in generating still more new output, as the string of binary digit symbols is computationally extended on the infinite DTM tape.

Turing's circle-free DTMs are tailored to accomplish for non-random algorithmically determined CSs what a denumerably infinitary application of the Dedekind-Peano successor function *S* is meant to achieve in sequentially stamping out the natural or counting numbers as a basis for elementary arithmetic. Where the Dedekind-Peano successor function generates the series, 0, $S(0) = 1$, $S(S(0)) = 2$, $S(S(S(0))) = 3$, etc., a similar rigorously definable function is needed for the real numbers, including irrationals, that are algorithmically computable.[5] Turing in his essay takes up the next natural challenge in working out the logical foundations of our knowledge of numbers. The natural numbers need only a denumerably infinite cumulative iteration of the successor function as their logical foundation, starting with 0 as basis, whereas real numbers, rational and irrational, that are not symbolized as possessing a random denumerably infinite digital sequence, need something more sophisticated to introduce these numbers to another more advanced development of arithmetic. Non-random infinitary digital sequences are the result of *computations*, and, as such, in Turing's concept, each CS stands in need of a complicated individual *machine*, a specific circle-free DTM.

Turing's circle-free DTMs, moreover, as a proper part of their output, recover all of the positive integers plus 0 that are cranked out by the Dedekind-Peano successor function *S*. Rational real numbers, 0, 1, 2, 3, etc., 1/4, 1/2, 3/4, etc., are among those whose CSs are the most easily computed, and for which it is easiest to suppose that there exists a circle-free DTM, effectively an algorithm. The iterative inductive application of the successor function can be thought of as like a machine turning out the natural numbers by applying to each successive value of the previous application of the function to its previous argument. Something like this mechanical model is needed for the computable or algorithmic as opposed to random real numbers, some of which like π are already known to exist. The circle-free DTMs are like the successor function, in that they can cycle endlessly, taking as new input whatever results from the previous application of the function to a different argument, in order to continuously generate new elements in an expanding set or series. The computable reals are just those for which there exists a circle-free DTM, as found in the appropriate algorithm involving the successor function to generate the natural numbers. The comparison is not meant to go farther than that. It follows that successor function *S* is computational, even though its iterative applications need not be understood mechanically as taking place over infinite ideal time.

The fact that there exists a circle-free DTM with the same CS as that of $\ldots S(S(S(0)))\ldots$, producing a denumerably infinite CS corresponding to any of the whole numbers, reinforces the independent assumption that the successor function by itself does not yield the

[5] Dedekind [1930–1932]; Peano (1889). The concept of the successor function itself is owing originally to Grassmann (1861). See Hilbert and Bernays (1934), vol. 1, 209.

most basic objects of arithmetic. Something needs to be done with the function over successive applications. We get the elementary numbers via succession only as the result of a denumerably infinite *operation*, which is to say a computation or calculation, involving the successor function, in which *S* is applied to 0, *and then S* is applied to *S*(0), *S*(*S*(0)), and so on, to denumerable infinity.[6] Otherwise, there is no *sequence* to the counting numbers, which itself implies something built up successively, and therefore over ideal if not actual time. This insight is the basis of Immanuel Kant's proposal in his [1781/1787] *Critique of Pure Reason* [*Kritik der reinen Vernunft*], that time as a pure form of intuition (perception) is the transcendental ground ultimately presupposed by our knowledge of arithmetic.[7] The Dedekind-Peano operation computes, for any iteration k of the successor function beginning after 0, a CS representing every individual whole number, among the other rational and computable irrational real numbers included in the set of all CSs. Turing expects there to exist a circle-free DTM that can symbol-edit a CS for any or all of the whole numbers, whose binary digital symbol strings, if we compare them with those of the far less internally repetitive CS for π, among others, are within relatively easy reach of the algorithmic symbol-editing that circle-free DTMs are supposed to provide.

The genius of Turing's concept is partly that his description of step-by-step mechanical symbol-editing instructions is based on a penetrating understanding of how human calculation would actually need to proceed at its essential most elementary level, for example, using pencil and paper.[8] The DTMs themselves are nevertheless automatic, as Turing insists, operating entirely on their own, without intelligent intervention. Loaded with a finite set of abstract machine instructions in their respective finite machine tables, the DTMs automatically undertake to construct their particular CSs. Their first instruction is to *begin*, with no need even for an invisible imaginary finger to hit the button '*Go!*'. It is no insignificant task to design DTM instructions in the rudimentary machine language that Turing details, as that to which all other DTM instructions are supposed to be reducible, and from which they can theoretically be computationally reconstructed. Consider only the instructions that would be needed for a DTM to print on its machine tape squares digit by digit the complex sequence of binary digits by which the irrational real number π, $\sqrt{2}$, or Euler's transcendental, extra-algebraic value *e*, is defined. What is done in practice for the sake of theory is to show how machine instructions can be prepared for the simplest cases, notably for whole or rational real numbers, and then maintaining that the same formal machine protocol must serve for any CS that is independently known to be computable by some circle-free DTM.[9]

Turing's circle-free DTMs symbol-edit their way algorithmically, over ideal infinite time, to produce all and only the CSs. They do so, even if, as automatic machines, in Turing's sense, they cannot be made to do anything more. Turing's concept of computability, like Alonzo Church's extensionally equivalent concept of 'effective calculability', published in the same year as Turing's (1936), in the papers, 'An Unsolvable Problem of Elementary Number Theory' and 'A Note on the Entscheidungsproblem', exactly fit the conceptual needs of mathematical logic and of still prevalent logicism in the foundations of mathematics, at this moment of convergence in its history with the beginnings of digital

[6] Wittgenstein (1922), 5.25, in presenting the *Tractatus* general form of proposition, insists on a comparable if not identical difference between function and operation.

[7] Kant (1965, §7, 79–82).

[8] See especially Copeland (2004b) for a detailed analysis of the concept and calculation of computable real numbers in a Turing machine context.

[9] An appreciative discussion of Turing's analysis of computation is given by Boolos et al. (2007).

machine theory under urgent external circumstances.[10] It is the need to supplement the successor function in Dedekind-Peano arithmetic with the rigorous counterpart for computable real numbers that justifies Turing in trying to discover what exactly would be required in order to produce the CSs of all and only the non-random real number sequences in their denumerably infinite binary digital expansions. By advancing a theory of the computable sequences and algorithmic digital symbol-editing machines, Turing, like Church in a parallel way, sought to bring non-random real numbers to arithmetic in a respectable fashion, like the natural numbers under the Dedekind-Peano successor function *S*, with only the formal syntax of a rigorously grounded logic at his disposal. Indispensable to the development of modern mathematics, the arithmetic of real numbers stands in need of something like Turing's concept of a constructive digital CS string-building circle-free DTM.[11]

## 3 Denumerablility of Dedicated Turing Machines

Turing pronounces on the cardinality of the CSs, and by implication on the cardinality of circle-free DTMs, in these terms:

> To each computable sequence there corresponds at least one [machine] description number, while to no description number does there correspond more than one computable sequence. The computable sequences and numbers are therefore enumerable.[12]

It might nevertheless be thought to follow from Turing's (1936) reduction of the unsolvability of the *Entscheidungsproblem* to the standard machine description numbers of all and only circle-free DTMs, that we cannot know the cardinality of the computable real numbers or their corresponding CSs. If there is no decision method for mechanically distinguishing all and only the CSs from the uncomputable binary digital sequences, then how can we expect to know how many CSs there are, and, in particular, whether the CSs are collectively denumerable or nondenumerably infinite in cardinality?[13]

Turing backs up his assertion concerning the denumerable cardinality of the CSs with an intriguing argument. He develops over several preceding pages leading up to his conclusion above that there is a unique positive integer coding, a standard machine description number ('S.D.'), for every DTM, circle-free and circular alike. He explains this numerical referencing of DTMs as being constructible from each DTM machine instruction table, in somewhat the way that Gödel numbers, by a glossary of basic terms associated with natural numbers that are then used to uniquely numerically code any construction in the syntax of a symbolic logic.[14] The machine instruction table of each DTM is itself a finite symbol

[10] Church (1936a, b).

[11] A useful contemporary discussion of the role of Turing's DTMs in modern mathematics is offered by Epstein and Carnielli (2008).

[12] Turing (1936, 241).

[13] Ibid., 248.

[14] The instructions for a Turing machine described in its *machine language* in the form of a *machine table* represents a 'complete' machine or *m*-configuration, $q_n$. Add a scanned symbol to the *m*-configuration $q_n$, and the result, in Turing's (1936) terminology, is a more simply expressed but internally more structurally complex logical 'configuration', $q_n$,S($r$). A *complete configuration* is the number of the square scanned, the complete sequence of all symbols on the tape, and the *m*-configuration. An *m*-configuration is expressed in the example, $q_1S_0S_1Rq_2$, as a single line of a machine instruction table, which can be assembled by an

string, capable of being integer-coded as a rather large number, and can accordingly be calculated as a CS by a DTM. There are only denumerably infinitely many integers. If the circle-free DTMs are a proper subset of the totality of DTMs, and if there can only be as many DTMs altogether as there are positive integers by which each DTM is coded, then there can only be denumerably infinitely many DTMs. Hence, there can be at most denumerably infinitely many circle-free DTMs, from which it further follows that there can equally be only denumerably infinitely many CSs.[15]

Turing's machine description number argument for the denumerability of all DTMs and all CSs is impressive on first encounter, but suffers under further critical reflection. It might be objected that, where the possibility of diagonalization looms, we cannot blithely take it for granted that *all* DTMs can be integer-coded. After all, a complete CSL listing of all CSs is begging for just such a diagonalization to be formalized upon it as a starting-place with respect to the two-dimensional horizontal-vertical matrix of binary digits that any CSL presents. If there is at least as good an argument to show that there are nondenumerably (transfinitely) many circle-free DTMs, then we must run out of any denumerably infinite supply of distinct integer standard machine description numbers *before* we have integer-coded *every* DTM, once we consider the possibility of how we are to assign a standard machine description integer code to a CSL-relative *diagonal* DTM or DDTM.

If Turing's inference from the unargued possibility of denumerably integer-coding all of the DTMs were correct, then following the same reasoning by analogy we could offer a Gödel-numbering objection to logically undermine Cantor's original diagonalization argument for the existence of higher orders of infinity, the transfinite order of nondenumerable cardinalities. First, we Gödel-number all sentences of the form 'R is a real number', for any R in the Cantor diagonalization target listing of reals, included among the *reductio*-hypothetically denumerably infinitely many real number digital sequences. When Cantor's diagonally constructed real number CDR is defined relative to a particular target list RL of all Rs, we simply Gödel-number the sentence, 'CDR is a real number', under the same coding conventions as those previously followed for similar sentences involving all real numbers R in the Cantor diagonalization target list. Since Gödel numbers, like Turing's standard machine description numbers, are positive integers, it should follow, as in Turing's proof of the denumerability of the CSs, that there can only be denumerably infinitely many Gödel numbers. We now paradoxically infer that there can only be

Footnote 14 continued
obvious device like a semicolon to represent the entire machine tape at any stage of the machine's production, as in: $q_1S_0S_1Rq_2$; $q_2S_0S_1Rq_3$; etc. With uniform substitutions of *A*, *C*, *D*, *L*, *R*, *N* and instruction punctuation sign ';', for corresponding symbols in these individual machine instructions and machine descriptions, Turing arrives at a *standard machine description number* (S.D.) (see Note 15 below). This step is a systematic syntactical variation undertaken as preparatory to Turing's presenting the concept of a universal computing machine and mounting his argument for the unsolvability of the *Entscheidungsproblem*. The coding method is applied in the first instance to the 'satisfactory' standard description numbers of all and only the circle-free binary symbol-editing machines, and their output, understood as computable symbol sequences, CSs, or computable real numbers themselves, functions, predicates, or the like, for which the addition of a decimal point is needed. It is Turing's counterpart of Gödel-numbering, arithmetizing the syntax of a logical language prior to constructing a type-restriction-respecting provability diagonalization, in which an unprovability predicate is not asserted of another predicate, but of the numerical object representing the very same unprovability predication. Machine tables are more conveniently expressed in reducible form for the machine language meta-variables of a specific binary digital symbol-editing machine, in Turing's (1936), and are recoverable from what Turing refers to as their 'abbreviated' or 'skeleton' machine instruction tables.

[15] Turing's method of standard machine description numbering (S.D.) is explained in Turing (1936, 239–241).

denumerably infinitely many real numbers, despite Cantor's proof to the contrary, on the grounds that each of them, including all of Cantor's diagonals, is mentioned in a sentence that is denumerably Gödel-numbered by a distinct integer. Such an inference obviously cannot stand, unless we are prepared to reject Cantor's diagonalization argument on the set of all reals as not after all proving the nondenumerability of the reals, and in particular as not proving the nondenumerability of the irrational reals.[16]

A stronger argument for Turing's conclusion that there are at most only denumerably infinitely many circle-free DTMs and CSs is perhaps that if there were nondenumerably infinitely many circle-free DTMs, then their standard machine description numbers, the Gödel-numbering-like integer coding of all and only the circle-free DTM machine instruction tables, could not all be stored on a single Turing machine tape. We may assume such to be the case in any total circle-free DTM archival application by a super DTM or SDTM, executing its machine instructions, on a *universal Turing machine* or UTM platform of the sort Turing describes in 1936, §6. A DTM has a denumerably infinitely long tape of denumerably infinitely many discrete information processing squares, and Turing believes that there are denumerably infinitely many DTMs. There appears to be no credible reason, then, why all the integers coding DTM standard machine table description numbers could not be inscribed on a single SDTM, or circle-free DTM-comprehensive archival application of the UTM. The relevant codes do not need to be physically inscribed on a physical symbol-editing tape. The DTM, its machine head, tape, and whatever symbols are printed on the tape at any finite stage of its digital crunching, whether existing in a mathematical domain or merely heuristic, are all ideal if not abstract.[17]

## 4 Turing's Cardinality Paradox: Diagonalization of Computable Sequences

It is possible to demonstrate, contrary to the above denumerability requirement, that the cardinality of all and only circle-free DTMs and all and only CSs is actually *non*denumerable. Turing, unfortunately, is not absolved thereby of addressing arguments supporting the opposite conclusion, that the totality of circle-free DTMs is also at most denumerably infinite in cardinality. The collision of these two comparably powerful conclusions, after the second opposed horn of the dilemma is also collected below, results in what we shall call *Turing's Cardinality Paradox*.

The diagonalization of surjectively (many-one) correlated DTMs and CSs, respectively, is precisely that polemically explicated by Turing. He is right to reject diagonalizations in the form he considers. There is, however, another, structurally similar, but importantly different version of diagonalization for any CSL assumed to include all and only the CSs supposedly constructed by all and only the circle-free DTMs, that are also both assumed to be denumerably infinite in cardinality. We adopt a somewhat different symbolization for the supposedly complete and supposedly denumerably infinite set of all and only the CSs and all and only the circle-free DTMs, for which it is argued that Turing's own remedy to forestall diagonalizations among the CSs is unavailable. Turing proposes to expose this kind of diagonalization as depending on a 'fallacy'. It is maintained in what follows that, even if it is true that the diagonalization Turing considers is fallacious in just the way he says, there is another version of the same general style of diagonalization that does not commit the fallacy. Properly interpreted, the alternative diagonalization on any CSL is

[16] Cantor (1932, 180–183). See Jacquette (2002).

[17] Turing (1936, 241–246).

itself computable, implying that in Turing's world the DTMs and CSs can only be nondenumerably infinite in cardinality. The computable diagonalizations that are the output of a Turing-type diagonal symbol-editing function applied algorithmically to the two-dimensional matrix of binary digits in any supposedly complete and denumerably infinite CSL nevertheless prove that both the circle-free DTMs and their CSs are nondenumerably infinite in cardinality.

Turing offers a concise statement of the diagonalization principle in criticizing one especially frontal type of diagonalization on the surjectively correlated DTMs and CSs. At the very beginning of his 1936, §8, Turing anticipates the following version of a Cantor-inspired diagonalization applied to supposedly complete denumerably infinite listings of all and only the CSs, and by extension concerning all and only the circle-free DTMs:

> It may be thought that arguments which prove that the real numbers are not enumerable would also prove that the computable numbers and sequences cannot be enumerable…Or we might apply the diagonal process. "If the computable sequences are enumerable, let $\alpha_n$ be the $n$-th computable sequence, and let $\phi_n(m)$ be the $m$-th figure in $\alpha_n$. Let $\beta$ be the sequence with $1 - \phi_n(n)$ as its $n$-th figure. Since $\beta$ is computable, there exists a number $K$ such that $1 - \phi_n(n) = \phi_K(n)$ [for?] all $n$. Putting $n = K$, we have $1 = 2\phi_K(K)$, i.e. 1 is even. This is impossible. The computable sequences are therefore not enumerable."[18]

If Turing's remarks apply to the diagonalization that we are about to propose, then so should his solution. Turing argues that the diagonalization fails because it embodies the false assumption that diagonal $\beta$ is computable:

> The fallacy in this argument lies in the assumption that $\beta$ is computable. It would be true if we could enumerate the computable sequences by finite means, but the problem of enumerating computable sequences is equivalent to the problem of finding out whether a given number is the D.N. [machine or $m$-description number] of a circle-free machine, and we have no general process for doing this in a finite number of steps. In fact, by applying the diagonal process argument correctly, we can show that there cannot be any such general process.[19]

From the essay, it is hard to judge whether Turing means to deflect diagonalization as a friend or foe of circle-free DTMs. It appears in the original context that Turing would be happy to avail himself of the diagonalization argument he discusses in order to address the *Entscheidungsproblem*, if only it were not subject to the nuisance fallacy he identifies.[20] Later, Turing offers a quite different but still recognizable diagonalization that avoids the objection, but has a rather different purpose, and from which Turing proposes to deduce the mechanical unsolvability of the *Entscheidungsproblem*.[21] It appears, on the contrary, that if

[18] Ibid., 246.

[19] Ibid.

[20] This is presumably to be inferred from the fact that Turing offers a similar diagonalization after exposing the fallacy in the first application. Ibid.: 'The simplest and most direct proof of this ["that there cannot be any such general process…of finding out whether a given number is the D.N. [machine description number] of a circle-free machine"] is by showing that, if this general process exists, then there is a machine which computes [diagonal sequence] $\beta$. This proof, although perfectly sound, has the disadvantage that it may leave the reader with a feeling that "there must be something wrong". The proof which I shall give has not this disadvantage, and gives a certain insight into the significance of the idea "circle-free". It depends not on constructing $\beta$, but on constructing $\beta'$, whose $n$-th figure is $\phi_n(n)$.'

[21] See Boolos et al. (2007, 3–40).

a Cantor-style diagonalization on the CSs and surjectively correlated DTMs of the type to be described succeeds for a supposedly complete CSL of supposedly denumerably infinite cardinality, then Turing's concept of computability, of a CS and DTM, is landed squarely in a logical-mathematical paradox of domain size inconsistency.

Turing's objection presupposes that in projecting a diagonalized CS, β, relative to a listing of binary digital sequences, CSLα (simply α, in Turing's original symbolism), we must have already solved the *Entscheidungsproblem*, so as to know whether or not a given DTM is circle-free by having determined that its standard machine description number is 'satisfactory'. If Turing succeeds in showing that the *Entscheidungsproblem* is algorithmically unsolvable, then that fact would evidently be a weighty impediment to the exact use of diagonalization to which Turing objects. Turing's proof of the unsolvability of the *Entscheidungsproblem*, unfortunately, depends on the assumption that the CSs and circle-free DTMs are denumerable, and that is precisely the assumption challenged by a Cantor-inspired diagonalization on the CSs in any CSL. It begs the question against the possibility of computable diagonalizations in the present argument context to assume that the CSs and circle-free DTMs are denumerable in cardinality, in order to argue that there can be no diagonalization on any CSL resulting in the nondenumerable cardinality of the circle-free DTMs or the CSs they compute.

Furthermore, the diagonalization target for the argument is a listing CSLα, supposedly, of all and only the denumerably infinite CSs computed by all and only the denumerably infinite circle-free DTMs. Turing at first appears right to maintain that we cannot posit or project a listing of all circle-free DTMs or the integers (themselves also real numbers) that code the standard machine descriptions of all circle-free DTMs. To do so, we must have already solved the *Entscheidungsproblem* that we know in advance to be unsolvable. The reason is that we cannot know prior to solving the decision problem which sequences are computable by DTMs, and which are not. Thus, we cannot get started in a diagonalization by projecting a listing all and only the circle-free DTMs. This limitation renders β uncomputable, and, as such, not itself a CS. To proceed in any such fashion, and conclude from these assumptions that there are nondenumerably many CSs or DTMs, would certainly be fallacious, as Turing maintains.

The question is whether in order to be a *computable* diagonalized CS, β, or DCS, relative to a denumerable listing of binary computable digital sequences, CSLα, the *Entscheidungsproblem* for algorithmically distinguishing the standard machine description numbers of circle-free versus circular DTMs must have already been solved. Understating things, such a strong assumption can easily appear doubtful, if we reflect that a solution to the *Entscheidungsproblem*, as it applies to the DTMs, must be a completely general method of mechanically deciding of any random standard machine description number whether it is 'satisfactory' or not. Whether, that is, the standard machine description number codes for a circle-free rather than circular DTM, and ultimately, in David Hilbert's original sense of the decision problem, whether any random mathematical statement is or is not a theorem of a given fully articulated mathematical axiom system. Whereas, all that need be argued in proving that there are nondenumerably infinitely many circle-free DTMs and corresponding output CSs, is that there exists at least one diagonal CS or DCS that cannot belong to any *reductio*-hypothetically complete and denumerably infinite listing CSL of all and only the CSs that are output by all and only the denumerably infinitely many circle-free DTMs.

Diagonalization arguments, and, in particular, the one about to be proposed, can also function in another way, with assumptions made at another level. Turing argues that if the sequences belonging to α are computable, then a computable diagonal operation on the

sequences in α is also possible, and in this, once again, he is certainly right. What does not follow is the inverse conditional that Turing by his reasoning needs in order to block the diagonalization, that if the sequences belonging to α are *not* computable, then a diagonal operation on the sequences is *not* possible. The denumerable circle-free DTMs and CSs might be surjectively correlated in any supposedly complete listing, here α:

CSLα
DTM1-0011001100110011...
DTM2-0101010101010101...
DTM3-1101101101101101...
⋮

Less strident assumptions than those Turing adopts are now minimally needed to support a diagonalization on the CSs belonging to CSLα. It need only be supposed, as Turing declares, that there are at most denumerably infinitely many CSs, and that these are surjectively correlated with denumerably infinitely many circle-free DTMs. We do not already need to know which in particular of all the logically possible DTMs are circle-free in order for this type of diagonalization to be described. The CSs by definition are computable, and we can project a surjective correlation of the *reductio*-hypothetically denumerably infinite complete listing of all and only the CSs with the set of all and only the circle-free DTMs that collectively compute all and only the CSs. We can consider the CSs freely in any random listing CSLα, moreover, or randomly reorganize and redistribute them, shuffle them around in any way we like, much as Turing already suggests.

We next define a diagonal DTM or DDTM on any CSLα, taking as its ⟨basis⟩ *any* $\mathrm{DTM_k}$ occurring in any row k (≥1) in CSLα, CSLα[DDTM⟨$\mathrm{DTM_k}$⟩], and outputting a diagonal CS or DCS relative to CSLα. A DDTM is a DTM that computes an *interpretation* of *another subordinate* DTM; or, more precisely, the DDTM scans and implements the machine instructions of the interpreted DTM, so as algorithmically to modify the CS1 of the interpreted DTM in a particular way, resulting in another CS2 than the subordinate DTM would have computed on its own. What follows, accordingly, is the most direct and obvious application of Cantor's diagonalization to the CSs, and, by implication, to the DTMs. The algorithm for DDTM, adopting some of Turing's 'abbreviated' or 'skeletal' machine instruction shorthand, the effect of DDTM⟨$\mathrm{DTM_k}$⟩ in CSLα[DDTM⟨$\mathrm{DTM_k}$⟩], is computed according to the following traditional Cantor-inspired diagonalization style function, reading $\mathrm{MI(d_k)} = \mathrm{P(0)} \rightarrow \mathrm{P(1)}$ as saying, conditionally, that if $\mathrm{DTM_k}$ is machine instructed to print 0 at digit place k in the CS it computes, then DTM DDTM⟨$\mathrm{DTM_k}$⟩ is to print 1 (instead)[22]:

DIAGONAL D-RULE

$$\mathrm{CSL}\alpha[\mathrm{DDTM}\langle \mathrm{DTM_k}\rangle = \mathrm{DTM_k}[\mathrm{P(d_1)}, \ldots, \mathrm{P(d_{k-1})};\ \mathrm{MI(d_k)} = \mathrm{P(0)} \rightarrow \mathrm{P(1)};$$
$$\mathrm{MI(d_k)} = \mathrm{P(1)} \rightarrow \mathrm{P(0)};\ \mathrm{P(d_{k+1})},\ \mathrm{P(d_{k+2})}, \ldots]]$$

[22] Turing (1936, §4, 235–239) explains 'abbreviated' or 'skeleton' machine table instructions. Turing, 237, indicates that he uses the arrow → 'to signify "the machine goes into the *m*-configuration…"'. The arrow in DIAGONAL D-RULE and UNIVERSAL INVERSION I-RULE instructs a DDTM to convert what the machine instructions of a subordinate DTM would have it print universally or in a particular symbol space on a machine tape into the contrary binary digit. This is certain to enter an *m*-configuration for the DDTM, in a command specifically instructing the conditional printing of a symbol. See Webb (1980) on computable Gödel numbering functions, by analogy with the computability of Cantor diagonalizations and the proposed diagonalization on the supposedly complete supposedly denumerably infinite listing CSL of all CSs.

Any CS is computable, by virtue of being a CS, and as such is logically guaranteed to be identical to the CS of at least one particular circle-free DTM. There is always at least one single circle-free DTM to print any CS we may happen to discover or find it natural to think about only more indirectly. Thus, there must exist a DTM to print the complements in succession of all the binary digits constituting the digital expansion of π, which we may find inaccessible unless we imagine systematically uniformly reversing the digits of π from 0 to 1 and 1 to 0. Such a procedure occurs whenever we project an algorithmic overriding interpretation of a subordinate DTM1, or an application on a UTM platform to the machine instructions of a DTM1 stored on the relevant DTM2 or UTM machine tape, in the form of its machine table, coded as its standard machine description number, and thereby subject the subordinate DTM1 to a potentially constant or potentially overriding CS-modifying algorithmic interpretation by DTM2.

There are unlimitedly many nondiagonal examples of the same basic interpretational computing structure. The pattern is to have one superordinate DTM applied to another subordinate DTM, where the interpreting DTM and interpreted DTM can both appear on the same machine tape of a UTM universal Turing machine platform. An example, building on the content of CSLα above, is DTMN⟨DTM1⟩, where CS(DTM1) = 010101…, and CS(DTMN⟨DTM1⟩) = 101010…; and where, for some DTM, CS(DTM) = CS(DTMN⟨DTM1⟩) = 101010… DTMN⟨DTM1⟩ algorithmically interprets the CS of DTM1 by inverting every digit in CS(DTM1) from 0 to 1 and 1 to 0. DTMN is, therefore, a *maximally diagonal* DDTM.

If, as we expect, we can mechanically apply DTMN < DTM1 > to output an algorithmic modification to the CS in any row k of any listing CSL, even without algorithmically identifying digit place k in CS(DTM1), then DTMN is a diagonal DTM, a DDTM, for any circle-free DTM outputting any CS in any row of any CSL. The CSL in question can be any supposedly complete and supposedly denumerably infinite listing of all and only the CSs produced by all and only the circle-free DTMs, including Turing's list α, designated here as CSLα. DTMN is fully algorithmic, logically dependent, as it is, on the CS of another circle-free DTM, whose exact sequence of binary digits it inverts more universally than the DIAGONAL D-RULE, according to this more easily expressed generalized binary digital toggling:

UNIVERSAL INVERSION I-RULE

$$\mathrm{IDTM}\langle\mathrm{DTM}\rangle = \mathrm{DTM}[\mathrm{MI}(\mathrm{d_k}) = \mathrm{P}(0) \rightarrow \mathrm{P}(1);\ \mathrm{MI}(\mathrm{d_k}) = \mathrm{P}(1) \rightarrow \mathrm{P}(0)]]$$

The UNIVERSAL INVERSION I-RULE, while easily taken over for full-carpeting diagonalizations upon the CSs of any CSL, and while uninteresting in such applications as that of DTMN⟨DTM1⟩, when CS(DTM1) = 010101…, becomes more interesting in an application like DTM2⟨DTM1⟩, for which the CS(DTM1) = π, $\sqrt{2}$, Euler's *e*, and other computable irrational reals.

The universal digital I-inverses of these CSs presumably are also computable, just as are any real number CSs that might be inverted. Unless, for some reason, we become especially interested in particular inversions, it is hard to imagine offhand how we would approach the specification of a single DTM machine instruction table to compute the algorithmic symbol-editing I-inversions of more complex CSs, such as those of π, $\sqrt{2}$, *e*, and their ilk, *except* by first computing the CSs, which are mathematically understood, by appropriate circle-free DTMs, and then algorithmically I-inverting the binary digits in each CS digit place k, sequentially, from 0 to 1 and 1 to 0, as UNIVERSAL INVERSION I-RULE prescribes. We can also be more specific in choosing from among any finite choice

of $k_1$, …, $k_n$ digit places in a selected CS for reversal of digital values in an already computed symbol string.

If we are uncertain as to whether the DIAGONAL D-RULE provides an algorithmic way of modifying a CS in a CSL at just one digit space k, if we think that an automatic DTM might not be able to locate a specific digit in a specific digit space k among the binary digital linear output prescribed by another DTM's machine instruction table, then we can proceed more democratically, so to speak. We can custom design a DDTM on the UTM platform that scans the machine instructions for a subordinate DTM and overrides DTM's output with a derivative supervenient symbol-editing operation that follows, instead of DIAGONAL D-RULE, the universally diagonal UNIVERSAL INVERSION I-RULE. Such a machine-implemented rule takes every CS in every CSL, and algorithmically interprets its computation by means of the diagonal UNIVERSAL INVERSION I-RULE instead. It algorithmically, sequentially, and, in Turing's sense, computably, converts to its alternative binary value every binary digit in the CS's expansion as determined by the machine instructions of the subordinate circle-free DTM, and thereby diagonally modifies the CS output of every correlated circle-free DTM in any CSL.

Precisely the same situation obtains computationally where, relative to a particular listing of CSs, CSL$\alpha$, DDTM interprets $DTM_k$ in CSL$\alpha$[DDTM$\langle DTM_k \rangle$], altering the output that would otherwise result from $DTM_k$. It does so by overriding the $DTM_k$ machine instructions for printing the digit, not in every CS digit place, as in DTMN$\langle$DTM1$\rangle$, but specifically and exclusively in digit place k in CS(DDTM$\langle DTM_k \rangle$), the kth binary digit in the kth row of CSL$\alpha$ as CSL$\alpha$[MI($DTM_k(d_k)$)]. If $DTM_k$ is the number of a DTM even randomly surjectively correlated with a (necessarily computable) CS at any row k of any CSL, then it follows from Turing's definitions that the DTM in question is circle-free. Computable diagonalizations by DDTMs interpret the DTMs responsible for any CS in any listing CSL of CSs. We do not need to know which these are or how particular circle-free DTMs are surjectively correlated with particular CSs in a CSL, in order to make correct cardinality inferences about how many there are, denumerable or nondenumerable. The DDTM interprets the circle-free DTM that outputs a CS by algorithmically overriding, modifying the DTM's output, and substituting instead a supervenient output of its own, based on the DTM's instructions.

However, the DDTM interpretation of the circle-free DTM is specially defined so as to result in a CS that is diagonal to listing CSL, just as DDTM$\langle$DTM$\rangle$, itself a DTM, is constructed to be diagonal to CSL. If CSL is a supposedly complete denumerably infinite listing of all CSs with all circle-free DTMs by which the CSs are computed, then there must be nondenumerably many CSs and consequently nondenumerably many circle-free DTMs, whose cardinalities always sink or swim together.

Contrary to Turing's assertion that a 'fallacy' undermines diagonalization efforts among the CSs and DTMs, the proposed diagonalization does not require a solution to the *Entscheidungsproblem* for distinguishing the standard 'satisfactory' machine description numbers of all and only the circle-free DTMs. Any of the DTMs to which DIAGONAL D-RULE or UNIVERSAL INVERSION I-RULE can possibly apply are *already* among the circle-free DTMs that Turing supposes are surjectively correlated with all and only the *computable* sequences, the CSs, belonging to any logically possible denumerable listing in any CSL. These CSs and the circle-free DTMs by which the CSs are computed in Turing's theory, on all logically possible listings CSL of which the proposed fallacy-free diagonalization is based, do not need to be independently validated as circle-free, conditional on a logically unattainable algorithmic solution to the *Entscheidungsproblem*. It is enough for purposes of the argument that if the CSs are denumerable, then they can be collected in a

CSL, construed as a two-dimensional matrix of binary digits, and, by virtue of their computability as computable sequences, they are the outputs of particular but unspecified circle-free DTMs.

The sequence produced by $DTM_k$ is assumed to be computable, which is to say a CS, and hence, by the algorithmic interpretation of $DTM_k$ by another DTM prescribed by DIAGONAL D-RULE or INVERSION I-RULE, a DDTM, $DDTM\langle DTM_k\rangle$, is certain also to be computable, $CSL\alpha[CS(DDTM\langle DTM_k\rangle)]$. This is predicted, since, relative to $CSL\alpha$, $DDTM\langle DTM_k\rangle$ modifies the computable binary digital CS output of $DTM_k$ in every or in at least one precisely algorithmically specified digit space k. $DDTM\langle DTM_k\rangle$, for example, obeys all of the machine instructions for the computable $DTM_k$, with the single exception of the digit occurring in $CS(DTM_k)$ digit space k. There, the interpretation algorithmically modifies the output of $DTM_k$ to produce a CS, a DCS, that another single DTM could also produce, but that logically cannot belong to any row k in any $CSL\alpha$. $DDTM\langle DTM_k\rangle$ interprets $DTM_k$ by modifying $CS(DTM_k)$, changing it in case $MI(DTM_k)$ prints 0 at digit space k, so that $DDTM\langle DTM_k\rangle$ prints 1 at k; while otherwise $DDTM\langle DTM_k\rangle$ prints 0 at k.

Importantly, for present purposes, DCS and $DDTM\langle DTM_k\rangle$ are guaranteed to be *diagonal* relative to the *reductio*-hypothetically complete denumerably infinite CS listing $CSL\alpha$, by virtue of which the $CSL\alpha[CS(DDTM\langle DTM_k\rangle]$ is logically excluded from $CSL\alpha$. By diagonal construction, the $CSL\alpha[CS(DDTM\langle DTM_k\rangle)]$ is supposed, *reductio*-hypothetically, to occur at some, and yet logically cannot possibly occur at any, row k in $CSL\alpha$.[23]

## 5 Implications of CS Diagonalizations by DDTMs

The diagonalization implies, first, that the CSs themselves are not denumerably but nondenumerably infinite in cardinality. Second, that the cardinality of the totality of all circle-free DTMs, including DDTMs, satisfying Turing's requirements for being circle-free, is also nondenumerably rather than denumerably infinite. The diagonalization avoids committing the 'fallacy' to which Turing objects in less careful diagonalizations, because the CSs to be included in $CSL\alpha$ do not presuppose a solution to the *Entscheidungsproblem*. They are simply all and only the *reductio*-hypothetically denumerable CSs that Turing himself says are forthcoming as output from all and only the *reductio*-hypothetically denumerable circle-free DTMs.

All items in any diagonalization listing CSL are accordingly computable. They are by definition all and only the CSs. However many of them there are, we allow them to appear in any denumerable order CSL, in case efforts at imposing a specific order on the CSs should itself turn out to be uncomputable. Regardless of how the assumed totality of CSs are arranged in a CSL, if collectively they are denumerable, then each one must appear at a distinct row k for any denumerable value of k, where the value of k goes from 1 to denumerable $\infty$. For the same reason, since each CS in any CSL consists of a denumerably finite symbol string of 0s and 1s corresponding to all and only the computable real numbers, the concept of denumerability implies that we can algorithmically modify the kth digit in any CS in any CSL, and instruct a DTM2 to interpret the output of another DTM1 so that the resulting $CS(DTM2\langle DTM1\rangle)$ is identical to the CS(DTM1), *except* for the binary digit appearing in all or any choice of precisely specifiable digital places k. DTM2 is thereby a DDTM (and, consequently, a DDTM is a DTM, if we were previously in the least doubt), that logically cannot be entered at any row k of any CS in any CSL to which the

[23] Boolos et al. (2007, 16–22).

DDTM applies. It follows, by the argument that adapts Cantor's diagonalization to any CSL of all and only the CSs correlated with all and only the circle-free DTMs, that there are both nondenumerably many CSs and nondenumably many circle-free DTMs.

Turing cannot accept that the CSs and circle-free DTMs are of transfinite cardinality, because he adopts what is arguably the inadequately supported, and on its own certainly logically open, conclusion, that *all* DTMs can be integer-coded. It might be preferable to say, *especially if* Turing demonstrates the unsolvability of the *Entscheidungsproblem* applied to standard machine description numbers, *if*, that is, Turing is right in believing that he has demonstrated that there is no algorithmic singling out of all and only the 'satisfactory' numbers coding all and only the circle-free DTMs, that we cannot ascertain even and especially as a matter of mathematical principle that the circle-free DTMs can be denumerably integer-coded, unless or until we have independently ascertained that the circle-free DTMs are denumerably infinite. Otherwise, Turing's integer coding argument for the denumerability of the circle-free DTMs blatantly begs the interesting question with respect to the cardinality of all CSs and all circle-free DTMs.

What, to speak rhetorically, is the possible *distinct* integer code of the DTM such that $\mathrm{CS}(\mathrm{DDTM}\langle\mathrm{DTM}_k\rangle) = \mathrm{CS}(\mathrm{DTM})$, when *all* of the DTMs surjectively correlated with *all* the supposedly denumerably many CSs in a CSL have *already* been integer-coded? What integers remain left over to integer-code a DDTM, if all the CSs in CSL are determined, in some Turing machine world sense, at least logically if not actually or ideally temporally *prior* to any diagonalization being made upon the CSL? We can variously express these diagonalization results involving DDTMs, relating them to the cardinality © of the totality of DTMs and of the CSs, or to the denumerably infinite totality of natural numbers, NN (or whole numbers, integers or positive integers), for both the set of all dedicated Turing machines, DTMs, and the set of all CSs:

$$\forall \mathrm{DDTM}, \mathrm{DTM}[©(\{\mathrm{DDTM}\langle\mathrm{DTM}\rangle\} \cup \{\mathrm{DTMs}\}) > ©\{\mathrm{DTMs}\}]$$
$$\forall \mathrm{DDTM}, \mathrm{DTM}[©(\{\mathrm{DDTM}\langle\mathrm{DTM}\rangle\} \cup \{\mathrm{DTMs}\}) > ©\{\mathrm{NNs}\}]$$
$$©(\{\mathrm{DDTMs}\} \cup \{\mathrm{DTMs}\}) > ©\{\mathrm{NNs}\}$$
$$©\{\mathrm{DTMs}\} > ©\{\mathrm{NNs}\}$$
$$©\{\mathrm{NNs}\} = \aleph_o$$
$$©\{\mathrm{DTMs}\} > \aleph_o; ©\{\mathrm{DTMs}\} = 2^{\aleph_o}$$
$$©\{\mathrm{CSs}\} > \aleph_o; ©\{\mathrm{CSs}\} = 2^{\aleph_o}$$

Turing's Cardinality Paradox can now be succinctly formulated as the inconsistency by which the DTMs are at once collectively in their totality denumerably *and* nondenumerably infinite in cardinality: $©\{\mathrm{DTMs}\} = \aleph_o \wedge ©\{\mathrm{DTMs}\} > \aleph_o$; $©\{\mathrm{DTMs}\} = \aleph_o \wedge ©\{\mathrm{DTMs}\} \neq \aleph_o$ (and similarity for $©\{\mathrm{CSs}\}$ by virtue of the surjective correlation of DTMs and CSs).

## 6 Observations, Reflections, Suggestions

These are among the dilemmas at the heart of Turing's Cardinality Paradox. The paradox, importantly, is not resolvable within Turing's canonical concept of a DTM. To challenge the nondenumerably infinite cardinality of the CSs and DTMs resulting from the listing-relative DDTM diagonalization is tantamount to casting doubt on the original diagonalization argument in Cantor.

Where Cantor assumes as a diagonalization target a listing of supposedly all denumerably infinitely many, or, more particularly, irrational, real numbers, the proposed diagonalization assumes a target listing CSL of all CSs. The result is the same for a denumerably infinite matrix of digits proceeding in denumerably infinite expansions in two directions away from a common origin. The claim that, under DIAGONALIZATION D-RULE (or UNIVERSAL INVERSION I-RULE), the DDTM$\langle$DTM$_k\rangle$ is a circle-free CS-computer, seems indisputable, if DTM$_k$ is a circle-free CS-computer and DDTM$\langle$DTM$_k\rangle$ interprets the CS of DTM$_k$ by altering it algorithmically, changing exactly one precisely specified digit (or exactly inverting any choice including all of them) that DTM$_k$ would otherwise print on its machine tape.

As requires more work to show, Turing's proof of the algorithmic unsolvability of the *Entscheidungsproblem* (whether for theorems and nontheorems or satisfactory and unsatisfactory standard DTM description numbers, or the like), does not go through, because Turing's argument logically depends on the assumption that the DTMs and CSs are denumerable. This is not to say that the decision problem *is* algorithmically solvable, only that there is a problem in Turing's effort to *prove* the contrary. With the collapse of Turing's proof of the unsolvability of the *Entscheidungsproblem*, the same applies, although again it takes more work to show, and for fundamentally the same reasons, to the so-called Halting Problem, inspired by Turing's (1936) essay, and originally proposed by Martin Davis, in his 1958 book, *Computability and Unsolvability*.[24]

If there is no satisfactory solution to Turing's Cardinality Paradox, then the very concept of a DTM and the CS it computes must at least provisionally be regarded as logically incoherent, the complete sets of each being both denumerable and nondenumerable in cardinality. Failing a satisfactory solution to Turing's Cardinality Paradox, Turing's or the Turing-inspired arguments made on behalf of the concepts of a DTM, circular or circle-free, and CS, the *Entscheidungsproblem* and Halting Problem, must be regarded as (classically) logically trivial, deductively implied by conjointly logically inconsistent assumptions. Concluding that Turing's proof must be unsound does not resolve Turing's Cardinality Paradox, or absolve Turing of the need to explain how the set of all circle-free DTMs is not after all both denumerable and nondenumerable in cardinality. There might exist, as a final teaser, an interesting logical-philosophical connection between Turing's Cardinality Paradox for the set of all circle-free DTMs and the logically more general semantic domain cardinality results established by the Löwenheim-Skolem Theorem and Skolem Paradox. As suggested in the final Sect. 7 below, the key to both Turing's Cardinality Paradox and the Skolem Paradox in understanding the Löwenheim-Skolem The-

[24] Davis (1982, 70). The literature on the Halting Problem is extensive. See, in particular, Anderson (1968), Herman (1969a, b), Boyer and Moore (1982), Rybalov (2007); Copeland (2010). The proposal to solve the Halting Problem for transfinite Turing machines, sometimes called Super Turing Machines or Infinite Turing Machines, is considered among others by Copeland (1998, 2002), Hamkins and Lewis (2000), Hamkins and Siebold (2001), Ylikoski (2005). Briefly, the *Entscheidungsproblem* and Davis's Turing-inspired Halting Problem applies the decision problem to computable functions rather than CSs. It states, on the strength of counterexample exceptions involving a diagonally defined function that halts if and only if it does not halt, that there is no DTM that effectively decides of any given DTM whether the DTM is circle-free or eventually halts its operations if once started on any logically possible machine tape with a particular symbol string.

orem is the dimensional geometry of a DTM's machine tape or binary digit symbol string.[25]

We may nevertheless question whether Turing's circle-free DTMs are needed to *explain* the *existence* of computable real numbers. If the answer is *No*, then perhaps, where the circle-free DTMs, if not the CSs, as Turing defines them, are concerned, we are dealing with nothing but an amusing and potentially pedagogically useful mathematical fiction. Perhaps Turing intended the DTMs as no more than a formal heuristic, although it seems a doubtful and even desperate way of reading the 1936 essay. If the answer is *Yes*, then we must ask, what explains the *existence* of the uncomputable randomly transcendental real numbers? These numbers also presumably exist, if computable real numbers exist, and their existence by definition does not depend on the existence of any DTM, circular or circle-free. What essential relation, in that event, does the presumed Turing world *existence* of any particular circle-free DTM bear to the *existence* of any particular computable real number binary digital sequence or CS? If the existence of uncomputable reals does not logically depend on their uncomputability, then the existence of computable real numbers should equally not logically depend on their circle-free DTM-computability. The ontic status of the real numbers generally appears to be logically independent of their computability or uncomputability, even for structuralists and constructivists in the philosophy of mathematics.

The situation would be different if mathematics regarded the uncomputable reals as nonexistent by virtue of being uncomputable. Understandably, however, there seems to be no enthusiasm for such an interpretation in the mathematical and philosophical literature. The uncomputable real numbers are just as existent as the computable ones, taking their ontic interdependence as one will. If the existence of the uncomputable reals does not depend on their uncomputability, then, without special argument, the existence of the computable reals should not be thought to depend on their DTM-computability. The existence of both computable and uncomputable reals alike cannot depend on the existence of circle-free DTMs, but on something logically and ontically independent of the DTMs, for which the DTMs at most suggest a method of exact descriptive identification and, obviously, computation or calculation. The CSs are especially important for unlimitedly many mathematical applications, and the formal protocols by which they can be computed

[25] Löwenheim (1915) and Skolem (1920). The suggestion is that the two conflicting positions in Turing's Cardinality Paradox might (a) be resolved in a way, if on the basis of Löwenheim-Skolem we can model any truth about nondenumerably many CSs and circle-free DTMs in a domain of only denumerable cardinality; although (b) since the semantic domain cardinality metatheorem also gives rise to the so-called Skolem Paradox, it might be suggested instead that there are deep conceptual confusions in the assumptions whereby Turing seems committed to both the denumerability and nondenumerability of the set of all and only CSs and the set of all and only circle-free DTMs. The problem is compounded if the only way out of Turing's Cardinality Paradox lands the mathematics of CSs and circle-free DTMs in the Skolem Paradox. The suggestion in the following Sect. 7 appeals to the possibility of accommodating computable diagonalizations or DCSs in linearly ordered denumerably many 3D cells, rather than 2D DTM machine tape squares, without ascending to machine tapes with nondenumerably many symbol-editing 2D squares or 3D cells. We can have nondenumerably many CSs and nondenumerably many circle-free DTMs, on the proposal, just as we can have semantic domains containing nondenumerably many entities in Löwenheim-Skolem. Although the machine information site where a CS digit is actually produced by any of the hypothetically nondenumerably many circle-free DTMs is itself either a 2D or 3D, it is not in any case nondenumerably dimensional in the length, width or breadth, even if continuous in extension, within its totality of discrete symbol-editing spaces. The analogy, finally, is that the facts about nondenumerably many CSs and circle-free DTMs, as in the Löwenheim-Skolem theorem, including the constructability of a DCS by a DDTM relative to a CSL can be modeled by a concept of a circle-free DTM, all of whose constitutive components are denumerable in dimensionality.

are at the heart of contemporary computing theory. Their existence conditions nevertheless appear logically disconnected from those of the DTMs, as are therefore their respective cardinalities. What is DTM-computability supposed to offer the foundations of mathematics, as distinct from computing theory and practice, beyond a captivating image of the origins of non-random real number sequences? If we do not actually need DTMs in order to accept the existence of CSs, then we must ask whether the concept of Turing's DTMs actually adds anything essential to our understanding of the theory and ontic status of CSs.

## 7 Higher Processing Dimensionalities and the Computational Mind

Whatever else they are, Turing's DTMs are ideal devices. There are no infinitely linear machine tapes or other computer storage media for DTM or UTM algorithmic symbol-editing in the world of physical entities. If there were, given the laws of physics, over infinite time, the beginning of the machine tapes would steadily be lost to the forces of entropy before the printing of infinitely many other symbols could actually occur further down the same tape at infinite distances from a tape's origin. Computers and animal brains, insofar as they approximate the idealizations of abstract DTM information processing, do not perform as perfectly or for infinite or indefinite potentially infinite spans of real time as the DTMs might be imagined to operate in Turing world ideal space and time.

DTMs as ideal entities in contrast must have the abstract mathematical and especially cardinality features we logically require of them, whether our purpose in considering DTMs is to make a contribution to the ontology of mathematics, or merely to provide a curious insightful heuristic to certain real number sequences. If the existence of computable diagonalizations on *reductio*-hypothetically complete lists of all the supposedly denumerably infinitely many CSs listed in any and every logically possible CSL bursts the symbol-editing bounds of linear information processing, then the suggestion here is to allow that Turing's machine tapes actually exploit the algorithmic symbol-editing potentialities of a more capacious densely packed three-dimensional space of denumerably infinitely many discrete algorithmic symbol-editing processing and printing spaces.

Imagine a machine table for DTMs with 3D machine tapes (3D-DTMs). These geometrically more capacious symbol-editing spaces can accommodate the processing of computable diagonalizations or DCSs relative to any CSL without succumbing to Turing's Cardinality Paradox. We can acknowledge, then, that the cardinality of all CSs and all DTMs is nondenumerably infinite. Denying that the CSs and DTMs are denumerable at once breaks the contradiction in Turing's Cardinality Paradox, and there is no contest unless we continue to assume, as we do in the case of the canonical 2D-DTMs (or minimal 1D Morse code-like counterparts, to which Turing also alludes), that algorithmic symbol-editing to produce CSs can only proceed along a linear two dimensional machine tape of denumerably infinitely many discrete symbol-editing squares. Turing can be supported in arguing that the cardinality of the CSs and circle-free 2D-DTMs must be denumerable. We might nevertheless imagine a succession of connected cubes, blocks or *cells* of information instead, moving along a conveyor belt beneath a DTM machine head, in place of Turing's linear sequence of discrete symbol-editing *squares*. The 3D-DTM machine head is directed to scan and symbol-edit the cells in ideal time as the three-dimensional content of a chunky 3D rather than the flat DTM machine tape consisting only of 2D symbol-editing squares.

The advantage of a 3D-DTM is that, alongside even a linear 2D symbol-editing progression on the same machine tape, there can also be stored the binary digits of another CS, and hence in principle of a DCS that is diagonal to any given complete listing CSL of all

CSs. There is enough space in a cell, as opposed to a square, not only for all the linear symbol-editing processing that Turing describes, but also for the parallel step-by-step construction of a CS that in its full glory turns out to be diagonal to any *reductio*-hypothetically complete listing CSL of the CSs produced by 2D machine tape DTMs, which mechanical operations on CSLs represented within the 3D cells must also be capable of computing. The CSs diagonal to any *reductio*-hypothetically denumerably infinite listing CSL of all and only CSs produced by all and only canonical DTMs with linear 2D machine tapes of denumerably infinitely many discrete symbol-editing squares fall as DCSs outside of the relevant linear ordering. They can have a comfortable home all along on the same symbol-editing medium, if we upgrade from Turing's 2D tapes to 3D-DTM machine information cells, in another related symbol-editing channel that is algorithmically constructible within the volume of symbol spaces afforded in a linear succession of 3D machine cells.

A DTM machine head, moving across consecutive machine cells rather than tape squares, can lay down symbol tracks for infinitely many different CSs within the sequence of machine tape cells, standing in many different logical relations to one another. Nor need such a machine head face an infinite number of tasks all at once in the first cell before it can proceed to its symbol-editing responsibilities in the second, third, and so forth, cell. The machine head can move along after some finite number of editings in *E*-spaces for symbol-editing processing in the first cell, moving on to the second, and, in ideal infinite time, returning at some point to make further editorial changes even within the *E*-spaces in the first cell, just as a canonical 2D-DTM can be instructed successively to return to the first *E*-squares on a 2D machine tape. Infinite time over denumerable symbol-space constraints is extremely forgiving of even the most astronomically complicated infinitely extended algorithmic zigzagging symbol-editing processes.

A computing model, in which multiple simultaneous CS buildings are possible within the same 3D cellular machine tape with a sufficiently intricate program, begins to approach something more like the mathematical complications presented by the information processing sites within the densely interfolded surface cortex of an animal brain. The biological computing machine should also be capable of processing information across any linkage of neuronal cells in its vast network or web in parallel fashion rather than as inadequately modeled by a canonical 2D-DTM. The brain manages this information processing traffic in a variety of different ways and at different information cellular levels, where a variety of interconnections are possible among components of signals moving in proximity within three-dimensional space, exploiting different geometrical relations in three-dimensions that are manifestly not available in two or less. If the mind is the brain and the brain is a symbol-editing machine plus peripherals, then the fact that we human beings have formulated and can understand the conservative adaptation of Cantor's original diagonalization to all the logically possible CSLs suggests that we, and surely then also intelligent nonhuman animals more generally, are *not* $\leq$2D *linear* UTM-platformed DTMs. Given the brain's 3D dimensionality, 3D-DTMs or applications of a 3D-UTM, DTMs or applications of the UTM with a 3D machine cellular linear ordering of 3D symbol-editing information storage and processing chambers, would seem to offer the more appropriate, dimensionally isomorphic model, of natural reasoning, rather than $\leq$ 2D information processing squares. The computable diagonalizations argument for any CSL forces this recognition of the need for greater information processing space, opening the way at the same time to many other explanatory advantages.

Such a space topologically permits us to track denumerably infinitely many computable diagonalizations or DCSs to any and all of the CSs in any and all CSLs, without demanding

expansion to 4D, 5D, etc., computing spaces, that may appear to exceed human computational capacity. It should also be possible to develop formal notations that permit certain computations about what happens in information processing spaces at comparative high altitudes of abstraction, without actually being able to specify how the relevant operations are supposed to be performed, even by an ideal nD-DTM. If, however, such higher-dimensional machine processors are actually needed for modeling the algorithmic specification of all CSs, if we can be pushed further and further in that direction by higher-order diagonalizations, then, since Turing's DTMs, as we said above, are ideal devices anyway, regardless of whether they are intended as ontic or heuristic, we may as well allow that their machine information units have whatever, even nondenumerable, dimensionality they might be thought required to have in order to keep pace with all of the ways in which computable diagonalizations or DCSs are algorithmically projectible as the symbol-edited output of higher and more highly stratified DDTMs.

The diagonalization hypothesis is that there are denumerably infinitely many CSs that can be enumerated in denumerably infinitely many different ways in denumerably infinitely many CSLs. Thus, the diagonalization argument only authorizes denumerably infinitely many computable diagonal CSs or DCSs. There should therefore be enough symbol-editing space for all of them within 3D-DTM machine symbol cells, as there are not, diagonals included, among the $\leq$2D canonical DTM denumerably infinite linear machine tape squares, or still more streamlined storage and processing media. The claim is not that animal brains are 3D-DTMs, but that 3D-DTMs are better formal models of animal brain information processing than $\geq$2D canonical DTMs with their $\geq$2D machine tapes. If 3D-DTMs suffice for all the denumerably infinitely many computable diagonalizations supported by any two-dimensional CSL, then there should be no justification for ascending beyond 3D to 4D, 5D, etc., whatever these further allowances might be imagined to mean for a 3D symbol-editing machine.[26]

## Appendix: Petzold on Turing's Cardinality of Machine Tape Blanks

Interpretations of Turing on questions of infinity and the cardinality of CSs and DTMs are compounded when Charles Petzold, in his 2008 book, *The Annotated Turing: A Guided Tour Through Alan Turing's Historic Paper on Computability and the Turing Machine*, recommends substituting 'now' for Turing's 'not' in an otherwise unremarkable passage in Turing's classic essay.

Turing (1936), in §6, *The universal computing machine*, 242, writes: 'In each complete configuration the blanks [on a machine tape] would all have to be replaced by "*D*", so that the complete configuration would not be expressed as a finite sequence of symbols'. Petzold (2008), 145, comments:

> The letter *D* represents a blank square. Turing doesn't want any breaks to appear in the complete configurations. He wants each complete configuration to be an unbroken series of letters. Turing's phrase, "so that the complete configuration

[26] The motivation for such an expansion in the dimensionality of machine tape symbol-editing spaces is lacking, because Cantor's diagonalization cannot be used to produce more than nondenumerably many DCSs or DDTMs. We must begin with a *reductio*-hypothetically denumerable CSL of all and only the CSs in order to establish that there are DCSs relative to CSL. There is no clear sense to be given a nondenumerable CSL, which in the first place is not a listing L, in order to prove that there are more than nondenumerably many CSs.

would not be expressed as a finite sequence of letters," is not quite clear. I suggest the word "not" should be "now". Certainly he doesn't want an infinite series of *D* symbols to represent a blank tape. Each complete configuration is finite.

Petzold's argument is not entirely convincing. It could be instead that Turing in the relevant passage is making precisely the literal observation as it occurs in the original text, leaving 'not' as we find it there, rather than replacing it as Petzold recommends with 'now'. The correct answer depends on how many blanks there might be on a DTM's machine tape at any stage of the DTM's execution of its instructions in algorithmically editing and printing infinitely many binary digital symbols on the tape. Turing, for all that Petzold shows, might equally have meant to say that since there are certain to be infinitely many blanks on a canonical DTM machine tape, the syntactical substitution procedure he describes with respect to an entire length of DTM machine tape at any stage of its symbol-editing operations could not be limited to finitely many *D*'s.

Moreover, there are several ways in which Turing might reasonably be understood to have intended the observation that are not compatible with substituting 'now' for 'not' in the relevant passage. First, Turing writes (1936), 233, that a DTM starts on a blank tape. The machine tape must nevertheless contain denumerably infinitely many linearly sequenced discrete symbol-editing spaces or squares, from which it seems only reasonable to deduce that any DTM machine tape will not merely have finitely many blanks in it, and that in abstracting its UTM counterpart there must be substituted infinitely many symbol *D*'s. Why should it not be so? The configurations, of which there must be denumerably infinitely many anyway, in order for any circle-free DTM to print out the denumerably infinitely many binary digits in the *F*-squares that constitute a complete CS, are not the same as the DTM's machine instructions, which, by Turing's definition, and for compelling independent reasons, can only be finite. Why, however, should the complete configurations be finite? How, indeed, *can* they be merely finite? Turing (1936) says explicitly, 235: 'The convention of writing the figures only on alternate squares is very useful: I shall always make use of it'. For Turing, it is not merely a luxury of convenience for his machines to have blank spaces to work with in bringing its apparatus mechanically to the stage of printing a 0 or a 1 in the alternating spaces of the machine tape provided for that main purpose. Circle-free DTMs cannot function without the computing workspace that exists in the only place it could possibly be found for such machines. Alternating *E*-squares and *F*-squares are indispensible. There cannot be infinitely many such squares prior to printing the first digits of the expanding CS produced by the operations of the circle-free DTM, because the DTM's finite symbol-editing instructions need to operate upon the results of previous executions of the DTM's program. Nor can the symbol-editing calculations be made after the printing of a DTM's respective CS on the same machine tape, since there is no 'after' to a CS, which, in the case of a computable irrational real number, continues building upon its digits to infinity. The DTMs must use calculating scratchpads in whatever blank spaces remain on the machine tapes, at any stage of a circle-free DTM's infinitely iterated symbol-editing operations, which can only be in the *E*-squares in between each pair of CS building *F*-squares, where the results of *E*-square computations are recorded as the circle-free DTM engages in calculations to output the next digit to be inscribed in the next *F*-square of the CS under construction.

Turing, continuing the thought of previously quoted 235, now writes: 'I shall call the one sequence of alternate squares *F*-squares and the other sequence *E*-squares. The symbols on *E*-squares will be liable to erasure. The symbols on *F*-squares form a continuous sequence. There are no blanks until the end is reached.' The *E*-squares are empty

calculating space. They divide the *F*-squares in alternating occurrences, as Turing describes their configuration on the machine tape. Processing takes place on the *E*-squares, and printing or inscribing digits take place on the *F*-squares. Once a digit is printed on an *F*-square by a DTM, it is never erased. That would be to uncompute a computed digit in the expansion of the CS that is under construction, which makes no sense in Turing's framework. To do so, moreover, would only invalidate whatever symbol-editing computations may have since been made assuming the stability of that value in the relevant *F*-square. Rather, we are to understand that the empty *E*-squares are used for all calculation space, where printings and erasures can take place ideally in indefinite cycles for the sake of calculating the next binary digit value in the *F*-squares of the ideal DTM machine tape representing the CS to the construction of which the DTM is dedicated. The 0 or 1 printed in an *F*-square is never erased, having been calculated, because to do so would imply a mistake in the previous computation, which one supposes ideally cannot happen. 'There is no need to have more than one *E*-square between each pair of *F*-squares,' Turing explains, again on 235. '[A]n apparent need of more *F*-squares can be satisfied by having a sufficiently rich variety of symbols capable of being printed on *E*-squares.' It follows that erasure is a function that applies only to the *E*-squares and never to the inherently progressive succession of binary digits being printed on machine tape by the DTM as representing if not actually constituting the DTM's dedicated CS.

The question is whether such an infinitely expansive computably symbol-printed machine tape is part of a complete configuration. The machine tape, in an obvious sense, is not part of the configuration itself, but is rather the product of an infinite succession of the machine's operations entering into one complete configuration after another, according to the machine's instructions. Turing explains the concept of a complete configuration in terms of these ideal time-laden operation stages, 232: 'At any stage of the motion of the machine, the number of the scanned square, the complete sequence of all symbols on the tape, and the *m*-configuration' will be said to describe the *complete configuration* at that stage.' If, equally obviously, we exclude complete configurations containing infinitely expansive symbol-printed machine tapes, then we have no way formally to abstract a sufficiently powerful universal computing machine description to mimic the output of a given DTM.

The circle-free DTMs and applied UTM accomplish their tasks by following a finite set of instructions with successively novel self-generating input over infinite ideal time. If an infinite symbol-printed machine tape does not exist as the result of a DTM's operations, then the DTM in Turing's sense can hardly be described as circle-free. An interpretation is thereby easily defended, under which Turing (1936), 242, in the passage Petzold discusses, might have meant 'not', as originally printed, rather than 'now', as Petzold proposes. The difference is immense, especially in our understanding of Turing's interrelated concepts of a computing machine, universal computing machine, machine or *m*-configuration, and complete configuration. The point is not to criticize Petzold for venturing a bold if lightly-argued suggestion intended to improve our understanding of the text by positing an imaginable typo. Petzold's proposal in the final analysis might after all be correct. The purpose is rather to emphasize the kinds of hazards besetting efforts to make univocal consistent sense of everything in Turing's (1936), particularly where cardinality issues are concerned, even after allowing for the liberal sprinkling of mistakes that Turing himself later corrects [1937], and that other commentators such as Post (1947), Kleene (1952) and

Davies et al. 2004, have identified and sought to explain and sometimes repair, or at least to help contemporary machine theory find its way around.[27]

## References


Anderson, M. (1968). Approximation to a decision procedure for the halting problem. *Notre Dame Journal of Formal Logic, 9*, 305–312.

Boolos, G. S., Burgess, J. P., & Jeffrey, R. C. (2007). *Computability and logic* (5th ed.). Cambridge: Cambridge University Press.

Boyer, R. S., & Moore, J. S. (1982). *'A mechanical proof of the unsolvability of the halting problem', ICSCA-CMP-28, 1–21*. Austin: University of Texas Institute for Computing Science and Computer Applications.

Cantor, G. (1932). *Gesammelte Abhandlungen mathematischen und philosophischen Inhalts, mit erläuternden Anmerkungen sowie mit Ergänzungen aus dem Briefwechsel Cantor-Dedekind, herausgegeben von Ernst Zermelo*. Berlin: Verlag von Julius Springer.

Church, A. (1936a). An unsolvable problem of elementary number theory. *American Journal of Mathematics, 58*, 345–363.

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

Copeland, B. J. (1998). Super Turing machines. *Complexity, 4*, 30–32.

Copeland, B. J. (2002). Accelerating Turing machines. *Minds and Machines, 12*, 281–301.

Copeland, B. J. (Ed.). (2004a). *The essential Turing: Seminal writings in computing, logic, philosophy, artificial intelligence, and artificial life, plus the secrets of enigma*. Oxford: Oxford University Press.

Copeland, B. J. (2004b). Computable numbers: A guide. In Copeland (Ed.), [2004a], pp. 5–216.

Copeland, B. J. (2010). Deviant codings and Turing's analysis of computability. *Studies in History and Philosophy of Science, 41*, 247–252.

Davies, D. W., Turing, A., & Post, E. (2004). Corrections to Turing's universal computing machine. In Copeland (Ed.), [2004a], pp. 103–124.

Davis, M. (1982 [1958]). *Computability and unsolvability*. New York: McGraw Hill. Enlarged Edition. New York: Dover Publications, Inc., 1982.

Dedekind, R. (1888). *Was sind und was sollen die Zahlen?* Braunschweig: Friedrich Vieweg und Sohn.

Epstein, R. L., & Carnielli, W. A. (2008). *Computability: Computable functions, logic, and the foundations of mathematics* (3rd ed.). Belmont: Wadsworth Publishing.

Grassmann, H. (1861). *Lehrbuch der Arithmetik*. Berlin: Enslin Verlag.

Hamkins, J. D., & Lewis, A. (2000). Infinite time Turing machines. *The Journal of Symbolic Logic, 65*, 567–604.

Hamkins, J. D., & Siebold, D. (2001). Infinite time Turing machines with only one tape. *Mathematical Logic Quarterly, 47*, 271–287.

Herman, G. T. (1969a). The unsolvability of the uniform halting problem for two-state Turing machines. *The Journal of Symbolic Logic, 34*, 161–165.

Herman, G. T. (1969b). A simple solution of the uniform halting problem. *The Journal of Symbolic Logic, 34*, 639–640.

Hilbert, D., & Bernays, P. [1934 (2nd edition1939)]. *Die Grundlagen der Mathematik*. 2 Volumes. Berlin: Springer.

Hodges, A. (1983). *Alan Turing: The enigma*. New York: Simon & Schuster (Touchstone).


[27] I am grateful to my research assistant Jan Walker for invaluable discussions of Turing's essay and the concepts of Turing machines and computable sequences. The present essay has profited substantially from his criticisms and energetic dissent from the argument of previous versions. I have also profited from the suggestions for improvement of an anonymous journal referee. A much-condensed summary of the main argument was presented under the title, 'Cantor's Diagonalization and Turing's Cardinality Paradox', at the conference on Computing, Philosophy and the Question of Bio-Machine Hybrids (AISB/IACAP World Congress), Birmingham, UK, 2-6 July 2012, and published in three page summary in the congress proceedings, under the conference title, in *Computing, philosophy and the question of bio-machine hybrids*, edited by J.M. Bishop and Y.J. Erden (London: Society for the Study of Artificial Intelligence and Simulation of Behaviour, 2012) (http://www.aisb.org.uk; ISBN 978-1-908187-11-6), 21–23.

Jacquette, D. (2002). Diagonalization in logic and mathematics. In D. M. Gabbay & F. Guenthner (Eds.), *Handbook of philosophical logic*, 2nd edition, (Volume 11, 55–147). Dordrecht: Kluwer Academic Publishing.

Kant, I. 1965 (1929) [1787/1787]. *Critique of pure reason*. Translated by Norman Kemp Smith. New York: St. Martin's Press.

Kleene, S. C. (1952). *Introduction to metamathematics*. New York: Van Nostrand.

Löwenheim, L. (1915). Über Möglichkeiten im Relativkalkül. *Mathematische Annalen, 76*, 447–470.

Peano, G. (1889). *Arithmetices principia nova methodo exposita*. Turin: Fratres Bocca.

Petzold, C. (2008). *The annotated Turing: A guided tour through Alan Turing's historic paper on computability and the Turing machine*. Indianapolis: Wiley Publishing Co.

Post, E. L. (1936). Finite combinatory processes formulation I. *The Journal of Symbolic Logic, 1*, 103–105.

Post, E. L. (1947). Recursive unsolvability of a problem of thue. *The Journal of Symbolic Logic, 12*, 1–11.

Rybalov, A. (2007). On the strongly generic undecidability of the halting problem. *Theoretical Computer Science, 377*, 268–270.

Skolem, T. (1920). *Logisch-kombinatorische Untersuchungen über die Erfüllbarkeit oder Beweisbarkeit mathematischer Sätze nebst einem Theoreme über dichte Mengen. Videnskapsselkapets skrifter, I. Matematisk-naturvidenskabelig Klasse*, Vol. 6, pp. 1–36.

Turing, A. N. (1936). Computable numbers, with an application to the Entscheidungsproblem. In *Proceedings of the London Mathematical Society*, series 2, 42: 230–265; Correction, 1937, 43, 544–546.

Webb, J. C. (1980). *Mechanism, mentalism and metamathematics: An essay on finitism*. Dordrecht-Boston: D. Reidel Publishing Co.

Wittgenstein, L. (1922). In C. K. Ogden (Ed.), *Tractatus logico-philosophicus*. London: Routledge & Kegan Paul Ltd.

Ylikoski, A. (2005). The halting problem on finite and infinite computers. *ACM SIGACT News, 36*, 132–138.