Signos Filosóficos, vol. XXIII, no. 45, January-June, 2021, 54-73

Ambivalence in machine intelligence: the epistemological roots of the Turing Machine

Belen Prado
ORCID.ORG/0000-0002-7908-3739
Leuphana Universität Lüneburg
Deutscher Akademischer Austauschdienst (DAAD)
prado@leuphana.de

Abstract: *The Turing Machine (TM) presents itself as the very landmark and initial design of digital automata present in all modern general-purpose digital computers and whose design on computable numbers implies deeply ontological as well as epistemological foundations for today's computers. These lines of work attempt to briefly analyze the fundamental epistemological problem that rose in the late 19th and early 20th century whereby "machine cognition" emerges. The epistemological roots addressed in the TM and notably in its "Halting Problem" uncovers the tension between determinism and uncertainty, regarded here as the primal and inherent features of machine cognition.*

Keywords: machine cognition; *Entscheidungsproblem*; determinism; uncertainty; mathematical logic

Reception: 07/10/2020 **Acceptance: 24/11/2020**

Signos Filosóficos, vol. XXIII, núm. 45, enero-junio, 2021,54-73

Ambivalencia en la inteligencia de la máquina: las raíces epistemológicas de la máquina de Turing

Belen Prado
ORCID.ORG/0000-0002-7908-3739
Leuphana Universität Lüneburg
Deutscher Akademischer Austauschdienst (daad)
prado@leuphana.de

Resumen: La Máquina de Turing (tm) se presenta como el hito y el diseño inicial de un autómata digital presente en todas las computadoras digitales modernas de propósito general y cuyo diseño en números computables establece bases profundamente ontológicas y epistemológicas para las computadoras de hoy. Estas líneas de trabajo intentan analizar brevemente el problema epistemológico fundamental que surgió a finales del siglo xix y principios del xx mediante el cual emerge la "cognición de la máquina". Las raíces epistemológicas que se abordan en la tm y, en particular, en su "*Problema de detención*" ponen al descubierto la tensión entre el determinismo y la incertidumbre, consideradas aquí, como las características primordiales e inherentes de la cognición de la máquina.

Palabras clave: cognición máquina; *Entscheidungsproblem*; determinismo; incertidumbre; lógica matemática

Recibido: 07/10/2020 **Aceptado: 24/11/2020**

"In answering the «Entscheidungsroblem», Turing proved that there is no systematic way to tell, by looking at a code, what that code will do.

That's what makes the digital universe so interesting, and that's what brings us here.

It is impossible to predict where the digital universe is going".

(George Dyson, 2012a)

Introduction

Finding an *universalis* algorithm that could mechanically compress the universe[1] using logical formalization was meant to be the final achievement pursued by philosophers and mathematicians in the late 19th and beginning of the 20th century. As a response to these attempts, the prototype of a synthetic mind emerged in 1936 and it is known today as the Turing Machine (tm) which constitutes the original and still-present design of basic cognition and computation processes in our day-to-day computers.

This article argues that the main epistemological breakthrough regarding mental activity and computation was triggered by Alan Turing's paper called: "On computable numbers, with an application to the *Entscheidungsproblem*" (1936),[2] which upholds that the prevailing tension between determinism and uncertainty constitutes the epistemological pinnacle of machine cognition. Turing's 1936 work integrates two fundamental aspects of machine cognition: on the one hand, the tm solves the *Entscheidungsproblem* (translated as the "decision problem") through what is known as "the halting problem". Turing unveiled the part

1 The mathematical understanding of the universe originally takes up Galileo's references by comparing it to a book "written in the language of mathematics" (Galileo, 1623, in Haugeland, 1985: 19) which is fundamentally based on a geometric and arithmetic view of the field. Also see Galileo's work "The Assayer" (1960: 151-336).

2 Despite the fact that Turing's article was published in 1937, it is mostly recognized from 1936. I will continue to use Martin Davis' references to this article as from 1936.

of mathematics that cannot be computed and demonstrated the non-existence of the algorithm that Hilbert and Ackermann (1928) asked for. On the other hand, it also establishes the ontological foundations of computers: the Universal Turing Machine (UTM), which ascribes to machines the possibility to recognize any other machine's encoded data and reproduce its behavior (given enough time, memory and embedding it into a specific language frame).

The first section of this article outlines the core epistemological crisis between mathematics and logic at the beginning of the 20th century while clarifying how the route through this crisis finally led to the TM's formulation. The second section is divided in two sub-sections; the first subsection describes in general terms the functioning of the TM for a broad understanding and contemplates the problem posed by countable and uncountable infinities. The second subsection underlines the notion of universality via the ontological state of the TM. The third section focuses on the ambivalence between determinism and uncertainty which the *halting problem* should help to clarify. This article will conclude that the latter concepts (that is to say, determinism and uncertainty) constitute the inherent features of the epistemological and ontological genesis of any digital computer.

The early historical and epistemological crisis

At the end of the 19th and beginning of the 20th century, many mathematicians and philosophers strove to explain the foundations of mathematics by peeling away its layers using formal logic instruments. This philosophical approach, best known as *logicism*, was fueled by the publication of *Principia Mathematica* by Whitehead and Russell in 1910. Its main representatives —Gottlob Frege, David Hilbert, Bertrand Russell, Alfred North Whitehead, Richard Dedekind— endeavored, though in different ways, to see mathematics as an extension of logic. The underlying question to prove whether logic could provide a consistent basis for mathematics led to one of the most significant epistemological crises in mathematical logic[3] (1850-1950) and it served as the embryonic stage of "machine

3 The foundations of mathematical logic begin with Aristotle's syllogistic logic and culminate in Boolean algebra whose extensional or set theoretical semantics does not imply the interpretation

intelligence" or "machine cognition", *i. e.*, what came to be called as the *Turing Machine*.[4]

At the bottom of mathematics, some *logicist* had mostly focused on the so-called *Hilbert's program* whereby all mathematical structures could be proved to be consistent (*i. e.*, the form A and not-A cannot be proven within the same formal system) through complete logical rules: "For Hilbert, noncontradiction was the essential condition of existence as such" (Hörl, 2018: 67). The search for a finite representation, a general method (*i. e.*, an algorithm)[5] in the language of a logical system was triggering the possibility that the whole universe of mathematics could be derived from Frege's system of logic,[6] namely first-order logic (*engerer Funktionenkalkül*).

Nonetheless, the intrinsic relation between symbols and logical rules underwent a systemic *déphasage*. It collapsed with Gödel in 1931, the Austro-Hungarian mathematician demonstrated that pure formal deduction could not encompass all

of symbols but how their fixed laws are combined. See, Boole, 2017 [c. 1854]. For a wider historical view of this passage from ancient logic to the modern one: see, Walicki, 2012.

4 Turing (1969 [c. 1948]: 410-432) uses the expression "logical computing machines" (LCM) instead of Turing machines.

5 In a German mathematical dictionary from 1747: *Vollständiges Mathematisches Lexicon,* the word "algorithm" was still referring to merely basic arithmetic operations. According to Petzold the modern use of the word algorithm (a finite set of steps able to perform a computation) only started to be used in the 1960's along with the "literature about computers" (2008: 41-42).

6 Frege (1993) is acknowledged as being the father of modern logic and of the basic rules of logical deduction that had been proposed in his work *Begriffsschrift* (1879), which are the first logical models of mathematical reasoning and also of natural language. These early formalizations were intended to be the steps towards a complete system as the one fostered by Leibniz, the *characteristica universalis* of which strove for universality fixed by logical calculus. Here, one can detect an "encyclopedia of human knowledge" operated by the indexicality of the calculation of symbols and with it the demonstration of true postulates. Finally overthrown by "Russell's Antinomy", Frege's system was nevertheless incorporated later by Whitehead and Russell in their *Principia Mathematica*. For more information see Davis, 2018.

mathematics, which he revealed through a sort of self-referential proof known as "Gödel's *Incompleteness Theorem*".[7]

After the setback produced by Gödel's Incompleteness Theorem, the British mathematician and philosopher Bertrand Russell (1872-1970) and the Hungarian mathematician John von Neumann (1903-1957) abandoned mathematical logic, yet the decision problem remained: was a given formula provable or not?

Finding a logical formal system able to encompass the roots of mathematics and thus to mechanically achieve a procedure of symbolically tracing "the full scope of human thought" (Davis, 2018: 11) had been the aim ever since the pioneer of the idea of universal automation, the *calculus ratiocinator*, Gottfried Leibniz (1646-1716) formulated the problem. But it was only two hundred years later when David Hilbert, along with his student Wilhelm Ackermann, made every possible effort to formalize it by means of the "*decision problem*".[8] A prosaic reformulation of the *decision problem* (1928) asks if it is possible to have a set of rules (an effective method) where you can input a mathematical formula and it consequently outputs whether it is true or not? That is to say, if there is any mechanical procedure that "decides whether any given number belongs to the set at issue or not" (Raatikainen, 2020). In short: is it possible to obtain a theorem from a given formula?

After Gödel's proof (1931) that logic, *i. e.*, formal axiomatic theories could neither provide a foundation nor produce all mathematics, the excitement about the *decision problem* dwindled. Although a solution for the *decision problem* could still exist, it would no longer demonstrate the truth of any theory because it either fails to prove its consistency or fails to prove its completeness, which are the necessary conditions for correct reasoning in a logical system.

7 The greatest epistemological crash proceeds from this aporetic state of axiom systems which either imply contradictions or can't even be proven within the system, this is what is known as "*Gödel's Incompleteness Theorem*": roughly, "the impossibility to determinate the truth of any given formula" (Petzold, 2008: 52).

8 This problem was outlined in the early 20th century at different Mathematical Congresses and later, concretized in a 1928's logic book: *Grundzüge der theoretischen Logik* (*Principles of Mathematical Logic*).

It is philosophically relevant not to skip over the epistemological problem posed at this point, which consists in the fact that Gödel showed that logical proofs cannot produce the full set of mathematical truth, but this does not entail that every true statement must necessarily have a proof to be true. To argue that something is "true" one must previously clarify the reference of the notion (*i. e.*, the distinction between a (syntactic) proof and (semantic) truth):

> A great deal of confusion can be caused by this, because people generally understand the notion of "proof" rather vaguely. In fact, Gödel's work was just part of a long attempt by mathematicians to explicate for themselves what proofs are. The important thing to keep in mind is that proofs are demonstrations within fixed systems of propositions. (Hofstadter, 1994: 26)

On the one side, the syntactic nature of truth requires proof: "You begin with axioms and derive theorems. Such theorems are said to be provable, meaning that they are a consequence of the axioms. With the syntactic approach to logic, it's not necessary to get involved with messy – possibly metaphysical – concepts of truth" (Petzold, 2008: 217). On the other side, the metamathematical and semantic nature of truth entails understanding the meaning[9] from the sentences at stake. "To Searle, this means that a digital computer – no matter how sophisticated it becomes – will never understand what it's doing in the same way that a human can" (Petzold, 2008: 347).[10]

The *decision problem* became the type of dilemma in the mathematical world that was ripe for a revolutionary leap: from the sunset of the logicist program[11]

9 In *Die Grundlagen der Arithmetik* (1884), Frege distinguishes between meaning (*Bedeutung*) and sense (*Sinn*), terms like "the morning Star" and "Venus" have the same meaning because they indicate the same object but nevertheless use different senses.

10 Some well-known arguments against the mind as algorithmic processes can be founded in: Nagel and Newman, 1958. Also, in Roger Penrose, 1989 and 1994. On the critical line about the efforts of mind-computer simulation, see; Searle, 1980. Finally, another argument against the view of minds as machines, Lucas, 1961.

11 Even though the logicist program has never been able to prove the impossibility of contradiction within its axiomatic systems, a logicist progress free of "Russell's paradox" (though not of rose the dawn of the most universal machine of all, the Turing Machine,[12] which marks the beginning of an epistemology for digital machines. As already stated, it was Turing who masterminded the "world crisis" in mathematical logic by finding that part of mathematics that can be run by a sequential and deterministic process we call "computers".

The Turing Machine

The serial machine

It wasn't until the spring of 1935 at Cambridge University in one of Max Newman's courses on the *Foundations of Mathematics* that Turing came to terms with the most prominent problems in mathematical logic, the proof of *Gödel's Incompleteness Theorem* and the still unresolved *decision problem*.

The proof for the non-existence of the algorithm incessantly sought by Hilbert was finally shown in Turing's paper (1936). This proof was not only a matter of abstract mathematics but also a bridge between "abstract symbols and the physical world" (Hodges, 2012: 93) because it opened the gates to the digital *kosmos*.[13]

At the foundation of the TM, phase-transitions of finite states in time (if-then transitions) are characterized as deterministic and serial procedures coded on a

all its axioms) had been possible with the "Zermelo–Fraenkel set theory" (ZFC) launched in the early 20th century. However, even when ZFC can prove the consistency of, for example, Peano arithmetic, the system is incapable for itself to prove consistency. See, Simpson, 1999. Also see Solovay, 1970.

12 Another formally equivalent algorithmic representation that can encode any sequential programming language can be found in "μ-recursive functions", or in the "λ-calculus" (lambda calculus), the latter was created by Alonso Church which shortly proceeds Turing's founding. This is the reason why Alan Turing had to briefly integrate the λ-calculus as a footnote in his 1937's paper, and so Church became Turing's Ph. D. supervisor. It was also Church who suggested to the logician Emil Post the unsolvable problem and whose results can be found in: Post, 1947.

13 I use the term "digital kosmos" for a general reference to the digital environment.

transition table where data is placed on a potentially infinite tape.[14] This mechanistic procedure is recognized as the fundamental iterative statement for the manipulation of formal and meaningless[15] tokens. The modern algebraist Charles Petzold states that the TM performs at a level so simple that "if the machine did anything less than what it does, it wouldn't do anything at all" (2008: vii).

This "essentially complex machine" (Blanco, 2013) is made up of three basic elements (tape, head, and program) which constitute what we recognize today in any electronic device under the names of memory and microprocessor, which reads and executes the programs. The scanning head reads the basic set of previously set-up instructions in order to adapt and modify its "internal state". For instance, *if* the internal state is 1, *then* move to the left and replace it with 0 and *then* move forward to the right and continue or break the process. Thereby, the machine adjusts its internal state by moving forward along the instructions chain until it produces the desired output and terminates.

The performance that corresponds to a value of a symbol (*scanned symbol*) is what Turing coined as; "m configuration", "internal states",[16] "mental states" which pictures the "mental image" that the machine scans according to a predefined set of rules, with the aim of keeping, generating or erasing a symbol: "The behavior of the computer at any moment is determined by the symbols which he is observing, and his <state of mind> at that moment" (Turing, 1936: 5). This *serial*[17] procedure (one instruction is executed after the other) from one "state of

14 The number of symbols contained on the tape (thus, the length of the tape) will depend on the computer's storage capacity. This extensional dimension constitutes the physical spatial aspect of the machine (*i. e.* the hardware), whose software, *i. e.*, the set of existing computable states or "states of mind", will be determined by the totality of all its possible combinations.

15 Symbols as such are meaningless, yet they become a meaningful symbol token through a fixed interpretation, it is not prescribed *a priori* what expressions it can designate. This arbitrariness pertains only to symbols (Newell and Simon, 1976).

16 Unlike TM, the "λ-calculus" does not have any internal state, the function that processes the mathematical operations works as a black box that does not afford to see the internal mechanics within it.

17 Daniel Dennett also offers a description of non-serial computers: "computers are serial [...]. There are exceptions; some special-purpose parallel-architecture computers have been created, mind" to the next one, limits the machine to observe only one box at a time while decoding the information and re-encoding it again on the tape.[18]

Each symbol represents a precise and finite value or set of values, as, from a computational point of view, a TM can only process one finite value at a time; a finite automaton. Consequently, a finite number of steps will allow obtaining a result, a process called "effective procedure". Hence, when the semantics of classical mathematics raised the problem of "infinite functions", it collided with what a TM can compute. The German artificial intelligence scientist Joscha Bach exemplifies: "Pi (π)[19] in classical mathematics is a value and is also a function, but it's the same thing. And in computation, a function is only a value when you can compute it. And if you can't compute the last digit of π, you only have a function" (2020: 16:00-16:15). In his 1936's paper, Turing uses the principle of parsimony to remove any mathematical excess: "We shall avoid confusion by speaking more often of computable sequences than of computable numbers" (1936: 233). The problem of non-computable numbers became clear for Turing by taking up Cantor's idea (1874) of the existence of real numbers as uncountable:

> In that same 1874 paper where Cantor demonstrated that the algebraic numbers are enumerable, he also demonstrated that the real numbers are not enumerable. [...] What Cantor eventually realized is that there are at least two kinds of infinity: There's an enumerable infinity and a non-enumerable infinity – an infinity of the natural numbers and an infinity of the continuum. (Petzold, 2008: 24-26)

but the computers that are now embedded in everything from alarm clocks and toaster ovens to automobiles are all serial architecture «von Neumann machines»" (Dennett, 2017: 155).

18 This decoding-encoding procedure is reminiscent of what the French philosopher Simondon (1924-1989) coined as *allagmatic* when he generally referred to cybernetics; "the general theory of exchanges and of *state modifications*" (Simondon, 1958, in Bardin, 2015: 15).

19 Charles Petzold describes how the construction of geometrical shapes is equivalent to solving certain forms of algebraic equations and traces it back to the ancient Greeks who thought it impossible to square the circle, they named this "obsessive activity" (τετραγωνίζειν) *tetragonize*. Just like the ancient Greeks, we can't construct a geometrical representation for the number π, since this irrational number "is not a solution to any algebraic equation" (2008: 13-35).

Therefore, on the one side, there is *computable infinity,* which can be defined as "real numbers that can be computed to within any desired precision by a finite, terminating algorithm" (Veisdal, 2019). For instance, real numbers like π might be considered "Turing complete or computable", that is, they can be executed through finite commands in a program through a process of approximation that will last indefinitely. As clarified by the Argentinian computer scientist Javier Blanco: "The original TMs were '*good*' when they did not finish and in the infinite process, they computed all the decimal places of some real number" (2020, personal correspondence). On the other side, there is an *uncomputable infinity* (which includes most of the real numbers); which are numbers not able to be computed at all and for which there is no algorithm that can compress and define them, so the only thing left to do is to write them digit by digit.[20] Hence, the functions computed by a TM are by definition computable functions: "Machines are definite: anything which was indefinite or infinite we should not count as a machine" (Lucas, 1961: 114). Even the functions that perform the simulated parallel processing done by artificial neural networks, that loosely mimics the human brain, are also replicated on these serial machines (Dennett, 2017: 155-156).

The universal machine

The universality of the TM is a consequence of the logical-mathematical interplay between the outside (symbols) and the inside (code number) for which the machine represents the outside that can be transferred to the inside: "a machine could be encoded as a number, and a number could be decoded as a machine" (Dyson, 2012a: 250).

20 One attempt has been made by the Argentine-American mathematician Gregor Chaitin in what is known as the "Chaitin's constant" (*Omega*) which is supposed to work as a solution to the halting problem. This questionable "halting probability" assumes that the sequences of random real numbers might be computable by a probabilistic solution. Here is an example of "Chaitin's Thought Experiment (Barmpalias, 2018): Suppose we run a universal Turing machine on a random binary program. Specifically, whenever the next bit of the program is required, we flip a coin and feed the binary output to the machine. What is the probability that the Turing machine will halt?" (Veisdal, 2019).

What is at stake in this type of universality is that any digital data is *ontologically* the same. Therefore, modifications of its internal states entail the adaptability of the machine's cognition. In fact, the chain of UTM essentially represents running a computer within another computer, *i. e.*, a computer that takes a program as data and runs itself. To put it simply, the idea of a machine copying another machine's behavior means that we give an input string (w) to the TM, and this one behaves by accepting, rejecting it, or looping through infinitely. To determine which is the machine's behavior < TM, w> we create another machine called UTM, which will receive as an input the TM with its string (w) and simulate the behavior of the host TM.

> Before Turing the general supposition was that in dealing with such machines the three categories – machine, program, and data – were entirely separate entities. [...] Turing's universal machine showed that the distinctness of these three categories is an illusion. (Davis, 2018: 143)

The digital *kosmos* is shrouded in universality. Just like the underlying foundations of analytic connections and their inferences of thoughts are to be found in the universality of the algebraic laws set-up by Boole, so is the foundation of the computer set-up by Turing profoundly rooted in the universal state of data:

> With the development of the second generation of electronic machines in the mid-forties (after the Eniac) came the stored program concept. This was rightfully hailed as a milestone, both conceptually and practically. *Programs now can be data, and can be operated on as data.* This capability is, of course, already implicit in the model of Turing: the descriptions are on the very same tape as the data. Yet the idea was realized only when machines acquired enough memory to make it practicable to locate actual programs in some internal place. (Newell and Simon, 1976: 117)

The ontology of machine cognition is generally based on pure *simulacra,* a fact that spurred Turing to develop in 1950 the simulation of the, later called, the Turing Test. The genesis of digital machines emerges with a machine able to behave as any other machine (UTM) and it follows the artificial intelligence idea of a machine that can behave as any given human being by capturing "the functional relations of the brain for so long as these relations between input and output are functionally well-behaved enough to be describable by [...] mathematical relationships [...] we know that some specific version of a Turing machine will be able to mimic them" (Sam Guttenplan, 1994: 595 in Copeland, 2020).

The ambivalence of machine intelligence

The ontological and epistemological commitments in philosophical and mathematical thought in the early 20th century was nothing like *φῐλοσοφῐ́ᾱ,* the love for wisdom, but rather an unending and uncertain path to the truth. Rephrasing Borges's poem: they "weave their incalculable labyrinth [...] not joined by love but by threat" (1974: 946).

The fundamental *structural* and *operational* ambivalence of the TM starts with the epistemological tension that took place when the operation of the learning process (the program's behaviors) run by the structure (set of predefined rules) produced inconsistencies in the process. "As Mathieu Triclot correctly points out, the fundamental problem is the statute of the rules, if they are fixed *a priori* or if they emerge as regularities from a learning process" (Blanco, Parente, Rodríguez, and Vaccari, 2015: 105). The aspect of the *uncertainty (Unbestimmtheit)* explicit in the halting problem emerges from a deterministic system (serial structure of predefined rules) operating at a high uncertainty level (program's behavior), which consequently does not allow to predict the machine's behavior.

The halting problem is "a proof by contradiction" that addresses the self-referential nature of the programs. It raises the question of whether there is a general-purpose program (GPP) that can always decide whether another program and its input will ever halt or not? Turing stripped the problem to its bare essentials: to prove the impossibility of the GPP, another program would have to be produced in order to use it. This second program will be the one that leads to a contradiction because there is at least one case (one counterexample) in which it cannot predict the behavior of the program being tested (GPP) since it might keep looping through *ad aeternum* or terminate its execution. Thus, the problem remains undecidable and the halting problem remains unsettled.[21]

21 The semantics of a logical system is based on what the terms of a theory refer to and the interpretation of its connectives, for a precise understanding of the undecidability of the semantics property (behavior) of programs see, Rice, 1953.

The nature of this problem is mainly self-referential: the GPP should be able to predict the behavior of any other program, and since the GPP is itself a program, it can be taken as an input for another program. The attempt to *determine* if it terminates or not and, at the same time, *being* a program itself leads into a contradiction. "Interestingly, the proof of this theorem shows that in any formal theory satisfying its conditions, one can write an analog of the liar paradox, namely, the sentence 'I am not provable in this theory'" (Walicki, 2012: 29). If true and provable, it leads to a false affirmation, and if false, there is something true that cannot be proven. The epistemological ambivalence in the domain of programs combines a perfectly determined machine (by knowing all of its instructions and inputs) with its unpredictable behavior because it lacks the formal methods to predict "what the behavior of a universal machine will be" (Blanco, 2014: 8).

> The impossibility of solving the halting problem mechanically means that its behavior cannot be predicted despite being deterministic; that the best that can be done is to carry out the execution of the machine for that particular case and see what happens, having to assume that sometimes nothing visible will happen, the machine will remain processing indefinitely. We can mechanically explain each step of the behavior of this machine without being able to know how it will behave globally. (Blanco, 2014: 8)

This fundamental tension affects machine cognition where it originates: the machine's *Umwelt* is neither to be found in its deterministic process nor entirely in the uncertainty that stems from the process but in their mutual mediation. The layout of the digital *kosmos* by means of machine cognition emerges from the epistemological ambivalence of its "mental acts": on the one side, the *sequential-determinist state* for which the TM uncovers the possibility to mechanically or effectively act according to predefined rules, and on the one side, the *uncertainty state* produced during the process itself. Ultimately, the reversed solution of the *decision problem* shown by Turing uncovers the fact that determinism does not entail predictability.

The structural level of logic-based rules generates efficient operations and self-regulations of computers through the rearrangement and combination of its datasets, obtaining results within a specific time frame. At the operational level, even when the machine's behavior is part of a predetermined system, it is not entirely governed by it. The *halting problem* formulates the uncertainty in the learning process of any computer, while the behavior of subsequent programmable systems is the result of this epistemological ambivalence.

This inherent indecision shapes the primal nature of every programmable system, and consequently pervades forthcoming uncertainty and leads to unexpected results in the learning process of the machines' behavior: "Turing's deterministic universal machine receives the most attention, but his non-deterministic oracle machines are closer to the way in which intelligence really works: intuition bridging the gaps between logical sequences" (Dyson, 2012b: 459-460).

Epistemological inconsistencies on the structural and operational levels (a completely deterministic machine does not allow formalizing the prediction of its output) constitute the grounds upon which all other synthetic cognitive architectures are based and whose far-reaching consequences entail more complex black box dilemmas.[22]

The joy in the evolution of machine cognition is driven by first principles "indifferent to their own truth" (Ortega y Gasset, 1958: 6) and whose axiomatization, far from securing a path to provable universality, opens up the ambiguity of programmable machines and their uncertain behavior.[23]

Conclusion

When Turing wrote his 1936's paper, he had in mind the idea of a "human being, equipped with pencil, paper, and time" (Dyson, 2012a: 247) and he proceeded by covering every state through the complete capacity of machine intelligence to compute until nothing human was left. The dawn of machine cognition accomplished

22 For the challenges regarding uncertainty in algorithmic trading or black-box trading and constructive alternatives, see, Stiegler, 2017.

23 Axiomatization is the self-consistent conceptual architecture that serves as the ultimate criterion of infallibility. Dyson sums the concept and its surrounding dilemmas up in the following terms: "Axiomatization is the reduction of a subject to a minimal set of initial assumptions, sufficient to develop the subject fully without new assumptions having to be introduced along the way" (Dyson, 2012a: 49).

the most important epistemological shift in history: a shift to the symbolic, and in doing so, laying the foundations for all subsequent learning processes.

The epistemological architecture of machine cognition was built on the aporetic ruins of Gödel's incompleteness theorem where the field of mathematics was left as fundamentally inconsistent or incomplete (the nature of self-referential systems is unable to prove to be formally true by finite means).

It becomes clear that this original tension between mathematical and symbolic reasoning plays a pivotal role in cognition. Even nowadays, the rate of uncertainty in programs is a fact that the probabilistic approach[24] of artificial intelligence corroborates. The main idea of intelligent behavior in machine cognition is expressed by the data compression achieved by "artificial intelligence systems" which roughly consist in the measured performances of combined "thought processes", "reasoning", and "behavior" techniques (Russell and Norvig, 2010).[25]

The origin of the TM meant a return to the root of the problem of the nature of mathematical reasoning, and although it remains unsolved, the TM pointed the way to digital automatization and served as the basis of more developed architectures of machine's cognition (e. g., symbolical, connectionist, hybrid architectures). All of them, even the most sophisticated ones, structurally operate within this original model of computable numbers which remains as the very foundation of any current machine's general learning process.

Acknowledgment

Special acknowledgment to the Deutscher Akademischer Austauschdienst (DAAD).

24 At present, probability as an approach for modeling algorithms in artificial intelligence is one of the most explored and used techniques and since this approach is not mainly guided by logical rules, it is subject to the imprecise subjectivity of belief that a given evidence will or will not be repeated in the future while reinforcing or undermining that belief. Also see, Pearl, 1988.

25 For a clear and technical definition of these three terms in the field of AI as an essential component of rationality, see especially the introduction of Russell and Norvig, 2010.

References

Bach, Joscha (2020), "Artificial Consciousness and the Nature of Reality | Lex Fridman Podcast #101", en *Lex Fridman*, June 13th, available on: [https://youtu.be/P-2P3MSZrBM&t=622s], accessed: October 7, 2020.

Bardin, Andrea (2015), *Epistemology and Political Philosophy in Gilbert Simondon*, London, Springer.

Blanco, Javier (2014), "Pensar y calcular", *Nombres. Revista de Filosofía*, no. 28, pp. 213-229.

Blanco, Javier (2013), "Una máquina que puede hacer todo", *Página 12*, April 17th, available on: [https://www.pagina12.com.ar/diario/ciencia/19-218171-2013-04-17.html], accessed: September 20, 2020.

Blanco, Javier, Diego Parente, Pablo Rodríguez y Andrés Vaccari (coords.) (2015), *Amar a las máquinas. Cultura y técnica en Gilbert Simondon*, Buenos Aires, Prometeo.

Boole, George (2017 [c. 1854]), *An Investigation of the Laws of Thought*, in *Project Gutenberg's*, available on: [https://www.gutenberg.org/files/15114/15114-pdf.pdf], accessed: Febraury 27, 2020.

Borges, Jorge Luis (1974 [c. 1964]), "Buenos Aires", in *Obras completas de Jorge Luis Borges*, Buenos Aires, Emecé Editoriales, p. 946.

Cantor, Georg (1874), "Ueber eine Eigenschaft des Inbegriffes aller reellen algebraischen Zahlen", *Journal für die Reine und Angewandte Mathematik*, vol. 77, pp. 258-262, available on: [http://www.digizeitschriften.de/main/dms/img/?PPN=GDZPPN002155583], accessed: June 6, 2020.

Copeland, B. Jack (2020), "The Church-Turing Thesis", in Edward N. Zalta (ed.), *The Stanford Encyclopedia of Philosophy*, Summer Edition, available on [https://plato.stanford.edu/archives/sum2020/entries/church-turing/], accessed: October 10, 2020.

Davis, Martin (2018), *The Universal Computer: The road from Leibniz to Turing*, New York, CRC Press Taylor & Francis Group.

Dennett, Daniel (2017), *From Bacteria to Bach and back: The Evolution of Minds*, New York, Penguin Books.

Dyson, George (2012a), *Turing's Cathedral: The Origins of the Digital Universe*, New York, Pantheon Books.

Dyson, George (2012b), "The dawn of computing", *Nature*, vol. 482, pp. 459-460. [https://doi.org/10.1038/482459a]

Frege, Gottlob (1993), *Logische Untersuchungen*, Göttingen, Vandenhoeck und Ruprecht Signatur.

Frege, Gottlob (1988 [c. 1884]), *Die Grundlagen der Arithmetik*, Hamburg, Felix Meiner Verlag GmbH.

Galileo, Galilei (1960), "The Assayer", in Galileo Galilei, Horatio Grassi, Mario Guiducci and Johann Kepler, *The Controversy on the Comets of 1618*, Philadelphia, University of Pennsylvania Press, pp. 151-336.

Gödel, Kurt (2006), "Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme I", *Monatshefte für Mathematik*, vol. 149, pp. 1-29. [https://doi.org/10.1007/s00605-006-0423-7]

Haugeland, John (1985), *Artificial Intelligence: The Very Idea*, Cambridge, Massachusetts/London, The MIT Press.

Hilbert, David and Wilhelm Ackermann (1928), *Grundzüge der theoretischen Logik*, Berlin, Springer.

Hodges, Andrew (2012), *Alan Turing: The Enigma*, Princenton, N. J., Princenton University Press.

Hofstadter, Douglas (1994 [c. 1979]), *Gödel, Escher, Bach: An Eternal Golden Braid*, New York, Basic Books.

Hörl, Erich (2018), *Sacred Channels: The Archaic Illusion of Communication*, Amsterdam, Amsterdam University Press.

Lucas, John Randolph (1961), "Minds, machines and Gödel", *Philosophy*, vol. 36, no. 137, pp. 112-127.

Nagel, Ernest and James R. Newman, (1958), *Gödel's Proof*, New York, Routledge.

Newell, Allen and Herbert Alexander Simon (1976), "Computer science as empirical inquiry: Symbols and search", *Communications of the ACM*, vol. 19, pp. 113-126.

Ortega y Gasset, José (1958), *La idea de principio en Leibniz y la evolución de la teoría deductiva*, Buenos Aires, Revista de Occidente/Emecé.

Pearl, Judea (1988), *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, San Francisco, Morgan Kaufmann Publishers Inc.

Penrose, Roger (1994), *Shadows of the Mind*, New York, Oxford University Press.

Penrose, Roger (1989), *The Emperor's New Mind*, New York, Oxford University Press.

Petzold, Charles (2008), *The Annotated Turing. A Guided Tour through Alan Turing's Historic Paper on Computability and the Turing Machine*, Indianapolis, Wiley Publishing Inc.

Post, Emil (1947), "Recursive unsolvability of a problem of Thue", *Journal of Symbolic Logic*, vol. 12, no. 1, pp. 1-11. [https://doi.org/10.2307/2267170]

Putnam, Hilary (1975), "The meaning of 'meaning'", *Minnesota Studies in the Philosophy of Science*, vol. 7, pp. 131-193.

Raatikainen, Panu (2020), "Gödel's incompleteness Theorems", in Edward N. Zalta (ed.), *The Stanford Encyclopedia of Philosophy*, Fall Edition, available on [https://plato.stanford.edu/archives/fall2020/entries/goedel-incompleteness/], accessed: November 07, 2020.

Rice, H. G. (1953), "Classes of recursively enumerable sets and their decision problems", *Transactions of the American Mathematical Society*, vol. 74, pp. 358-366.

Russell, Stuart and Peter Norvig (2010), *Artificial Intelligence: A Modern Approach*, New Jersey, Pearson Education.

Searle, John R. (1980), "Minds, brains, and programs", *Behavioral and Brain Sciences*, vol. 3, no. 3, pp. 417-457.

Simpson, Stephen George (1999), *Subsystems of Second Order Arithmetic*, Berlin, Springer.

Solovay, Robert Martin (1970), "A model of set-theory in which every set of reals is *Lebesgue Measurable*", *Annals of Mathematics*, vol. 92, pp. 1-56.

Stiegler, Bernard (2017), *Automatic Society*, Volume 1: *The Future of Work*, Cambridge, Polity Press.

Turing, Alan (1969 [c. 1948]), "Intelligent machinery", in Bernard Meltzer and Donald Michie (eds.), *Machine Intelligence*, vol. 5, Edinburgh, Edinburgh University Press, pp. 395-432.

Turing, Alan (1950), "Computing machinery and intelligence", *Mind*, vol. 59, pp. 433-460.

Turing, Alan (1936), "On computable numbers, with an application to the *Entscheidungsproblem*", *Proceedings of the London Mathematical Society*, vol. 42, no. 1, pp. 230-265. Republished (1937), vol. 43, pp. 544-546.

Veisdal, Jørgen (2019), "Uncomputable Numbers. Real numbers we can never know the value of", in *Medium*, available on [https://medium.com/cantors-paradise/uncomputable-numbers-ee528830d295], accessed: June 23, 2020.

Walicki, Michal (2012), *Introduction to Mathematical Logic*, Singapore, World Scientific.

Wolff, Christian von (1747), *Vollständiges Mathematisches Lexicon*, Leipzig, Gleditsch.

Belen Prado: Ph.D. student at Leuphana Universität Lüneburg, Germany, research area: philosophical aspects of Machine Learning. Preceded by undergraduate studies in Political Science and Philosophy, and postgraduate studies in Political Philosophy at the University of Buenos Aires, Argentina. These brief academic links constitute only one representative aspect of several years of work, research, cooperation, and participation in different academic and social fields related to the area of so-called "artificial intelligence".

D. R. © Belen Prado, Ciudad de México, enero-junio, 2021.