Minds and Machines (2024) 34:S29–S49
https://doi.org/10.1007/s11023-023-09634-0

True Turing: A Bird's-Eye View

Edgar Daylight[1]

Received: 18 May 2022 / Accepted: 10 April 2023 / Published online: 19 June 2023

Abstract
Alan Turing is often portrayed as a materialist in secondary literature. In the present article, I suggest that Turing was instead an idealist, inspired by Cambridge scholars, Arthur Eddington, Ernest Hobson, James Jeans and John McTaggart. I outline Turing's developing thoughts and his legacy in the USA to date. Specifically, I contrast Turing's two notions of computability (both from 1936) and distinguish between Turing's "machine intelligence" in the UK and the more well-known "artificial intelligence" in the USA. According to my proposed historical interpretation, Turing did not view computations in the real world to be exhaustively and deterministically characterized by his automatic machines from 1936.

Keywords Biography of Alan Turing · History and philosophy of computing · Turing machine · British idealism

1 Turing's Milieu

The British astronomer Stanley Arthur Eddington (1882–1944) was one of the first scientists to understand and disseminate the general theory of relativity of Albert Einstein (1879–1955) in the Anglophone world (Vibert, 2021). While Eddington regarded the universe as physically indeterminate, Einstein remained convinced that it was intrinsically deterministic: "God does not play at dice [with the universe]."[1] This dichotomy shall play a key role in the present article in connection with Alan Turing (1912–1954), who was eagerly reading Eddington's writings in late 1929.[2]

Karel Van Oudheusden" is the author's real name.

[1] Einstein's famous words are discussed on page 178 in van Dongen (2010).

[2] Including Eddington's *Internal Constitution of the Stars* (Eddington, 1920) and *The Nature of the Physical World* (Eddington, 1929a); cf. page 40 in Hodges (1983).

✉ Edgar Daylight
egdaylight@dijkstrascry.com

[1] School of Media and Information, Siegen University, Herrengarten 3, 57072 Siegen, Germany

Eddington, who would become Turing's mentor at Cambridge University, believed that beyond the symbols of the theoretical physicist lurks a mystical truth, which the modern scientist cannot grasp mathematically. As he wrote in 1929:

> Surely then that mental and spiritual nature of ourselves, known in our minds by an intimate contact transcending the methods of physics, supplies just that interpretation of the symbols which science is admittedly unable to give.[3]

Besides "symbolic knowledge," there was "intimate knowledge," which remained out of mathematical reach.[4] Enjoying a rainbow, laughing at a joke and playing a musical instrument all require intimate knowledge.[5] According to Eddington, there was an unbridgeable chasm between the universe in which we live (on the one hand) and logico-mathematical symbolism (on the other hand).[6]

A similar narrative came from the astronomer James Jeans (1877–1946). By late 1929, Turing had already started reading one or more books by Jeans.[7] The following excerpt from *The Mysterious Universe* (Jeans, 1930) captures Turing's intellectual milieu of idealism and free will:

> The terrestrial pure mathematician does not concern himself with material substance, but with pure thought. His creations are not only created by thought but consist of thought, just as the creations of the engineer consist of engines.
> To my mind, the laws which nature obeys are less suggestive of those which a machine obeys in its motion than of those which a musician obeys in writing a fugue, or a poet in composing a sonnet. The motions of electrons and atoms do not resemble those of the parts of a locomotive so much as those of the dancers in a cotillion.
> If all this is so, then the universe can be best pictured, although still very imperfectly and inadequately, as consisting of pure thought, the thought of what, for want of a wider word, we must describe as a mathematical thinker.[8]

Reading the passage in reverse, the "mathematical thinker" of Jeans was no machine; instead, he was comparable to a musician who writes a fugue. Likewise, nature was no automaton.

For Eddington, Jeans and other scholars in the twentieth century, scientific uncertainty replaced the Laplacian maxim that it is possible to analytically predict all future states of the universe from initial conditions and natural laws alone. A young Turing expressed a similar view of scientific uncertainty around 1932:

> It used to be supposed in Science that if everything was known about the Universe at any particular moment then we can predict what it will be through all the future. … More modern science however has come to the conclusion that … we are quite unable to know the exact state … As McTaggart shews[,]

[3] Quoted from pages 22–23 in Eddington (1929b).
[4] See pages 321–322 in Eddington (1928).
[5] Paraphrased from pages 22–23 in Eddington (1929b).
[6] See Eddington (1929b) and also the recollections of a contemporary (Dingle, 1954).
[7] See pages 40 and 51 in Hodges (1983).
[8] See page 124 in Jeans (1930).

> matter is meaningless in the absence of spirit … Personally I think that spirit is really eternally connected with matter but certainly not always by the same kind of body.[9]

Turing's reference in the previous passage to the Hegelian philosopher, John McTaggart (1866–1925), of Cambridge University aligns with his reception of the works of Eddington and Jeans. For McTaggart, "the recognition of a unity of a universe" was "greater than that recognized by ordinary experience or by science" (McDaniel, 2020). He held the view that "it is possible to be conscious of this unity in a way different from that of ordinary discursive thought" (McDaniel, 2020). McTaggart's idealism was, as the scholar Gerald Rochelle puts it, "not so far removed from the contemporary scientific world-view as we might at first think."[10]

Besides the writings of Eddington, Jeans and McTaggart, Turing was also aware of the work of Bertrand Russell (1872–1970) from the first decade of the twentieth century, a period in which Russell had promoted his logicism. His intent had been: "To discover a logically ideal language … that will exhibit the nature of the world in such a way that we will not be misled by the accidental, imprecise surface structure of natural language."[11] Russell believed that mathematics in its entirety could be grasped by logical rules: "The fact that all Mathematics is Symbolic Logic is one of the greatest discoveries of our age; and when this [so-called] fact has been established, the remainder of the principles of mathematics consists in the analysis of Symbolic Logic itself."[12] Mathematics could be completely reduced to a fixed, predefined logic, according to Russell.

Did Turing agree with Russell's outlook? Did Turing believe that all creative actions of "the" mathematician can be reduced to one and the same symbolic framework? Perhaps not, in line with the aforementioned view of Eddington et al. Perhaps not, due to the intellectual influence exerted by Cambridge University mathematician, Ernest Hobson (1856–1931), on a young Alan Turing.[13] While Eddington was stressing the distinction between symbolic and intimate knowledge in the 1920s, Hobson had already distanced himself from Russell several times a decade earlier, including with the following words: "Mathematics is a living and growing science" and "a mathematician is a human being, not a logic-engine" (Hobson, 1910). According to Hobson, "the" mathematician did not exist, and it is impossible to specify all future actions in mathematics in one fixed language. Russell's logicism had to give way. In a similar vein, there is historical evidence suggesting that Turing gave a lecture to philosophers at Cambridge University in 1933, in which he made it clear that Russell's logicism had shortcomings: "…

[9] The entire quote appears on pages 63-64 in Hodges (1983) and is archived under Turing, catalog number AMT/C/29, www.turingarchive.org/browse.php/C/29

[10] See pages 16–17 in Rochelle (2018).

[11] Quoted and discussed in Irvine (2017).

[12] Quoted from page 5 in Russell (2009).

[13] In my judgement, Zhao Fan (Fan, 2020) and Peter Millican (Millican, 2021) stand out as Turing scholars in that they already describe certain links between Hobson's writings on the foundations of mathematics and Turing's work.

a purely logistic view of mathematics was inadequate; and … mathematical propositions possessed a variety of interpretations, of which the logistic was merely one."[14]

According to W.J. Mander, author of *British Idealism: A History*, Russell and his colleague G.E. Moore (1873–1958) at Cambridge University produced "celebrated arguments against idealism" and "there can be no doubting their vigour" but "it must also be appreciated that, as the opening salvos of a war they went on to win, these attacks have been remembered as more powerful and decisive than they really were, either historically or philosophically."[15] Did Turing argue for or against idealism in the 1930s?

Britain's philosophical landscape had been predominantly idealist at the close of the nineteenth century.[16] To presuppose that 30 years later idealism was no more to be found (in, say, Turing's Cambridge) would be a mistake. Mander's idealists include McTaggart, Eddington and Jeans, but not Turing, hence my proposal for fellow Turing scholars to engage with Mander, particularly in connection with his opening statement:

> The movement is known as British *Idealism*, and here too we find a vital point of unity; a common affiliation—not to Berkeley but to Plato, Kant, and Hegel—which bound a generation together. In 1860 there were scarcely any idealists, by 1900 the majority of philosophers so designated themselves, but thirty years later they were rare again. Yet it will not do just to leave matters at that; for although they were all idealists, the philosophers to be studied were not all idealists in the same way. Indeed as the movement progressed there came to be developed a great variety of such positions, ranged against a set of outlooks they variously called empiricism, materialism, or dualism …, many incompatible with each other and some scarcely distinguishable from those of their opponents.[17]

In addition to philosophical inclinations, such as those of idealism and materialism, I shall also, from now on, explicitly distinguish between two groups of historical actors: Camp A versus Camp B. According to members of Camp A, the gap between the logical world of natural laws (on the one hand) and reality (on the other hand) can be bridged. It is merely a matter of finding those laws of nature. According to Gottfried Leibniz (1646–1716), Russell in the early twentieth century and several computer scientists in recent decades, we can grasp reality symbolically, once we have developed the right symbolic logic.[18] Moreover, that symbolism is even taken to be algorithmic among computer scientists today; it is then a matter of

[14] The philosopher, Richard Braithwaite (1900-1990), wrote down these words to summarize Turing's talk; see pages 85-86 in Hodges (1983).

[15] Quoted from page 544 in Mander (2014).

[16] See page 1 in Mander (2014).

[17] Quoted from page 5 in Mander (2014) with his emphasis.

[18] The rationale for placing Leibniz and Russell (circa 1910) into the same camp is that both men definitely did not belong to Camp B. However, see, e.g., Corry (2015) for more nuance with regard to the history of mathematics proper.

unraveling the universe's algorithm. According to Camp B, which includes members such as Eddington and Hobson, the gap is difficult if not impossible to bridge. It is out of the question if only one fixed, symbolic framework can be employed. Any pre-specified symbolic language will fail to capture reality in all its facets.

The specific members of Camp A will be of less concern in the remainder of this article than those of Camp B. Most computer scientists and believers in artificial intelligence are members of Camp A, while the same cannot be said of physicists and engineers.

Five historical interpretations, which I wish to put forward to historians, can now be conveyed in brief. Firstly, at the start of his university studies in Cambridge, Turing was well aware of both views A and B; moreover, just like Eddington et al. and Hobson, he was a fervent supporter of Camp B. Secondly, as author of his 1936 paper, "On computable numbers …" Turing adhered to the agenda of Camp A, but he did not believe in that agenda. Thirdly, in later years and especially after the war, Turing championed the viewpoint of Camp B, as my discussion of Turing's work on machine intelligence will reveal. Fourthly, unlike Turing and many physicists at the time, most computer scientists today do not consistently distinguish between both views. Computer scientists believe that all processes in the physical world are algorithmic (and thus are also logically determined). Fifthly, they wrongfully thank Turing for this algorithmic outlook on the universe.

I will largely substantiate these five points in my contribution, which consists of the present article and a forthcoming exposition. The present article provides a bird's-eye view on Turing's developing thoughts as an idealist and his legacy in the USA to date. My forthcoming narrative provides a worm's-eye view, revealing that Turing was an idealist both before and after World War II.

Does the word "idealist" in the previous paragraph refer to purely Platonic idealism or to Eddington's transcendental idealism? (Perhaps the word refers to one of Mander's many other guises of idealism.) A key difference between both mentioned forms of idealism can be conveyed by revisiting the following part of Jeans's excerpt:

> … the universe can be best pictured, although still very imperfectly and inadequately, as consisting of pure thought …

If Jeans meant to say that more research would allow scientists to acquire a complete, mathematical grip on the universe, then I take him to be a purely Platonic idealist. (The mathematics involved would have to be, say, probabilistic in order to account for the universe's uncertainty.) However, if Jeans believed that the mathematical picture of the universe would always remain imperfect, then his philosophical position aligned more closely with Eddington's transcendental idealism; Eddington proclaimed that symbolic knowledge was orthogonal to intimate knowledge. The latter could not be reduced to the former, not even in principle. Zooming in on Eddington's and Jeans's forms of idealism, Mander reports thus:

> [Eddington's] thought that the world we experience is but a symbol of some 'more behind' relates this idealism back to that of Carlyle, but its subjective scepticism owes more to Berkeley than either Kant or Hegel. Jeans too felt

> that quantum theory had brought us to a world 'very different from the full-blooded matter and the forbidding materialism of the Victorian scientist … But in so far as he regarded the new reality revealed as mathematical—'The Great Architect of the Universe now begins to appear as a pure mathematician' his creation 'more like a great thought than like a great machine' …—his was a more purely Platonic idealism.[19]

My preferred way to read Turing (anno 2023) is to view him as an idealist without further stipulation. I will not substantiate my working hypothesis that Turing was a transcendental idealist, that his philosophical position was far from consistent and that he sensed a potential inconsistency in his own Weltanschauung—which would explain why he was keen to learn from Ludwig Wittgenstein in 1939 and Dorothy Sayers in 1941.[20] Even someone of Russell's caliber was in a conceptual muddle in the 1920s, largely because of the advent of modern physics.[21] It should, therefore, come as no surprise that Turing's position was philosophically scrutinized at the time, as well as in later years.[22]

Since Andrew Hodges is Turing's leading biographer (Daylight, 2014), I shall repeatedly rely on his detailed findings.[23] In my reading, Hodges views Turing in the main as a materialist.[24] Since I propose that Turing was an idealist, the potential novelty of my contribution is immediate.[25]

2 Contentious Points in Mathematics

At the turn of the twentieth century, scholars disagreed about the consistency of specific mathematical measures taken in connection with infinitely large sets. Too much freedom in pursuing abstract mathematics easily led to contradictions.[26] It was, for example, unclear whether a mathematician was allowed to posit the existence of a set containing an infinite number of elements and subsequently use it in his mathematical argument, without providing further specifications of how the set could come about. Was it permissible for a mathematician to rely on an infinite number of throws of a fair die, when postulating the existence of a new set? Or was he always expected to specify its contents in a logically determinate manner? In Europe, Georg Cantor (1845–1918) and Godfrey Harold Hardy (1877–1947)

[19] Quoted from page 551 in Mander (2014).

[20] See page 211 in Hodges (1983) and Daylight (2021).

[21] See Chapter 3 in Joad (1932) for scrutiny of Russell's "neutral monism.".

[22] See Joad (1932) for scrutiny of Jeans and Eddington et al.; Shanker (1987) and Schmidt (2011) for a Wittgensteinian critique of Turing's seminal work.

[23] I apologize upfront for not engaging with more than a handful of Turing scholars. The secondary literature on Turing has become too vast.

[24] See page 103 in Hodges (1983) and page 530 in Hodges (2001).

[25] Part of the sequel to this introduction is based on a translation from Van Oudheusden and Dutré (2022); all rights remain with the present author.

[26] I use "set" as an umbrella term to sketch the general landscape, leaving "aggregate" and other terminology to Moore (1982).

belonged to the first group: almost anything was allowed, to put it simplistically, so long as one continued to reason consistently—but guaranteeing that consistency was precisely where the difficulty lay. Hobson was a member of the second group of logically determined sets. Russell was at first close to Hardy but gradually slipped to a position between Hardy and Hobson. These developments mainly occurred during the first decade of the twentieth century.[27]

Hobson was not principally against the idea of rigorously formalizing various branches of mathematics. He championed logical determinacy and the dictum that every infinitely large set had to be specified on the basis of a "norm," which meant that the arbitrariness obtained with an infinite number of throws of a fair die was not permitted inside each mathematical branch.[28] Hobson's concept of "norm" would later be refined by Turing, leading up to Alonzo Church's concept of "algorithm," which depended on the so-called "Turing machine" (Church, 1937). Today, the Turing machine is rightly or wrongly regarded by many computer scientists as *the* mathematical model of the modern computer.[29]

However, Hobson did not believe that mathematics could be captured once and for all. All creative actions of all mathematicians (including those yet to be born) were not specifiable in one predetermined language. Hobson expressed his disagreement with Russell on this point in the third part of his 1910 Address to the British Association for the Advancement of Science:

> It is quite impossible for me here to discuss … the interesting question of the possibility of setting up a final system of indefinables and axioms which shall suffice for all present and future developments of Mathematics. (Hobson, 1910)

Mathematics was a living organism, it evolved and could not be captured symbolically in advance, as, according to Hobson, Russell believed was possible. Hobson continued:

> After all, a mathematician is a human being, not a logic-engine. … Not every great mathematician possesses in a specially high degree that critical faculty which finds its employment in the perfection of form, in conformity with the ideal of logical completeness … (Hobson, 1910)

Mathematicians had to free themselves from Russell's logicistic shackle. In the second part of his Address, Hobson put the matter thus:

> The belief is very general amongst instructed persons that the truths of Mathematics have absolute certainty, or at least that there appertains to them the highest degree of certainty of which the human mind is capable … [A] considerable amount of difference of opinion of this character exists among mathematicians at the present time. (Hobson, 1910)

[27] See Chapter 2 in Moore (1982).
[28] See Chapter 2 in Moore (1982).
[29] See, e.g., Davis (2000) and Daylight (2015).

While "instructed persons," such as Russell, belonged to Camp A, Hobson was defending the views of Camp B.

Perhaps Turing reasoned in a similar vein two decades later, advocating a pragmatic take on symbolic logic (cf. "a purely logistic view … was inadequate"). If it is fair to presume that Turing was, pace Eddington, no advocate of a Laplacian worldview (let alone of an algorithmic universe) then perhaps Turing belonged to Camp B instead of Russell's Camp A. The irony is that, as author of his 1936 article, he merely played along in conformity with the logicistic agenda of Russell and, to be more precise, the formalism of the German mathematician, David Hilbert (1862–1943).[30]

3 The 1936 Article

Turing's pre-war contributions to modern logic were connected to the Russellian-Hilbertian developments of the second and third decades of the twentieth century. The title of Turing's 1936 article in full is: "On computable numbers, with an application to the Entscheidungsproblem." A discussion of the Entscheidungsproblem itself lies outside the scope of this article, but the words "On computable [real] numbers" echo the aforementioned disagreements between Hardy, Russell and Hobson with regard to the infinite number of digits constituting the representation of a real number.

Technically, Hobson had articulated his 1906 position concerning the admissibility of actual infinity thus:

> The process of arbitrarily choosing figures one after the other, without cessation, involves the idea of endlessness only, and this is quite distinct from the truly infinite process which can be regarded as defining a definite object. In the latter case the process regarded from outside is a completed one embodied in the law which dominates it; in the former case it is impossible to regard the process from the outside.[31]

The actual infinite was permissible for Hobson, provided the "process" at hand was "dominated" by a "law." Hobson did not believe, however, that *all* of mathematics could be dominated by a fixed, predefined body of laws.

Should historians view Hobson (rather than, say, Russell) as Turing's mathematical ancestor?[32] In my reading, the "computable numbers" of Turing in 1936 were real numbers with "computable" taking on one of two connotations: an A-connotation and a B-connotation:

[30] The distinctions between Russell's developing philosophical positions and the later Hilbert Programs, as well as the pivotal role that Heinrich Behmann (1891–1970) played in transferring ideas from Russell to Hilbert, will be deliberately passed over in this article. See Sieg (2013) for extensive coverage of the Hilbert Programs, Mancosu (1999) for details on Behmann and Corry (2008) for a nuanced description of Hilbert's personal aim.

[31] See page 28 in Hobson (1906).

[32] Turing referred on page 246 in Turing (1936–7) to pages 87–88 in Hobson (1921).

A. "automatic machines," in line with Hobson's 1906 technical exposition

> "… there is an axiom … which expresses the rules governing the behavior of the [human] computer …"[33]

B. "choice machines," in line with Hobson's helicopter view on mathematics as a living organism

> "… cannot go on until some *arbitrary* choice has been made by an *external* operator. This would be the case if we were using machines to deal with axiomatic systems …"[34]

The behavior of a disciplined human computer is, as expressed in case A, completely logically determined by an axiomatic recipe. In case B, however, that same computer may rely on creative—or, at any rate, non-predetermined—input from the outside. While the logicistic shackle of Russell is expressed in passage A, I suggest that Turing's overall take on mathematics was, pace Hobson, that "the" creative actions of "the" mathematician could not be reduced to one, a priori, fixed symbolic framework, i.e., Turing's brief mention of option B reveals his personal preference.

Following Hobson, my Turing favored a pragmatic attitude toward modern logic. He did not believe that the gap between the practices of creative mathematicians (on the one hand) and their formalization (on the other hand) could be closed by relying solely on his A-notion of computability. The ability to switch from one symbolism to another, yet-to-be-developed, framework remained inherent in human intelligence. Turing used his A-notion to demonstrate that Hilbert's finitistic stipulations were not powerful enough to solve the Entscheidungsproblem algorithmically.[35] The adverb "algorithmic" in the previous sentence refers to the A-connotation of computability, which Turing defined in 1936 on the basis of his "automatic machines" (including his universal automatic machines). Those machines were called "Turing machines" as early as 1937 by the logician Alonzo Church. Following Church and much of his post-World War II doctoral students, computer scientists today are very familiar with Turing machines, but most of them do not know about Turing's "choice machines" (option B).

This brings us to Turing's repeated usage of the word "machine" in his 1936 article and his machine metaphor, in general. His writings indicate that, for him, both a human being (e.g., a creative mathematician) and an actual device were, in essence, mathematical machines—or "machines" in short. However, each of these machines was not per se an automatic machine (option A), some of them could be his choice machines (option B) instead. Likewise, Turing did not categorically distinguish between symbolic machinery (such as his automatic machines) and energy-consuming machinery. In essence, they were all mathematical. To

[33] Quoted, with my emphasis, from page 254 in Turing (1936).

[34] Quoted, with my emphasis, from page 232 in Turing (1936).

[35] Turing showed that even if one resorts to Hilbert's Platonic realism, the Entscheidungsproblem is still unsolvable. So, it remains unsolvable inside Hilbert's finitistic program as well. I thank Erhard Schüttpelz for sharing this insight with me.

recapitulate, I characterize Turing as a mathematician who did not categorically differentiate between an abstract static world of universals and the concrete changing world of individuals.[36] Zooming out, it then comes as little surprise that those scholars who were tasked with interpreting and evaluating Turing's 1936 article—including his Cambridge lecturer (Max Newman, 1897–1984) and his future doctoral supervisor at Princeton (Alonzo Church, 1903–1995)—did not act in full accordance with Turing's thoughts. They were sympathizers, if not members, of Camp A and they were less familiar with Eddington's physical indeterminism and British idealism in general. Turing was thus the odd one out; on reflection, it is not surprising that he often felt misunderstood. In contrast to Turing and physicists of the likes of Jeans and Eddington, many of Turing's colleagues in both modern logic and computer building did persistently distinguish between abstract (non-causal) objects on the one hand and causal (spatiotemporal) objects on the other hand.

4 Machine Intelligence

Immediately after World War II and a decade before the advent of artificial intelligence in the USA, Turing became involved in the design of the Automatic Computing Engine [ACE] and subsequently programmed it to advance his personal "machine intelligence" project.[37] He wanted to use a post-war engineered machine to achieve intelligent behavior. However, his theoretical impossibility result in connection with the Entscheidungsproblem, along with those of Kurt Gödel and Alonzo Church also in the 1930s, showed that every disciplined (human) computer—i.e., every "automatic machine" from 1936, or every "Turing machine" or "algorithm" from 1937—is intrinsically limited in his/its computing power. That is, every type-A computing machine comes with problems which it cannot solve autonomously, while human intelligence leads to the insight just conveyed and, therefore, seems to be capable of accomplishing more. In Turing's words: "The human intelligence seems to be able to find methods of ever increasing power for dealing with such problems 'transcending' the methods available to machines."[38] The creative actions of "the" mathematician, if there were any such notion to begin with, could never be captured with a Turing machine.

To nip the (just voiced) criticism against machine intelligence in the bud, Turing turned to his second source of inspiration: Eddington's physical indeterminism. A creative mathematician, such as Carl Gauss (1777–1855), is not a disciplined mathematician, according to Turing.[39] From the perspective of Camp A, Gauss's actions were logically determined, despite the fact that Gauss had made mistakes in his mathematical practice—mistakes that, pace Turing, might even have been

[36] My proposal to fellow scholars is to find out how Turing fits in Mander's history of British idealism; see Mander (2014).

[37] See page 358 and onward in Hodges (1983).

[38] Quoted from page 411 in Turing (1948). Not only do these words refine Hobson's 1910 view pertaining to formalization, they also convey Turing's doctoral work (Turing, 1938).

[39] See page 411 in Turing (1948).

essential for Gauss to accomplish his mathematical feats.[40] If Gauss had been allowed to make mistakes in his mathematical work, why would the ACE machine not also be allowed to enjoy such freedom? The deployment of the ACE and other post-war machines had to be, so Turing insisted, brought in line with his B-notion of undetermined computability.[41]

According to Turing, the ACE need not be perceived as an A-machine, but rather, as an "interference machine" that continually interacts with its environment, like humans do, before exhibiting determined (and hopefully intelligent) behavior. In 1948, Turing wrote:

> [Man] is in frequent communication with other men, and is continually receiving visual and other stimuli which themselves constitute a form of interference … We shall now consider machines in which such interference is the rule rather than the exception.[42]

A student learns a lot by making mistakes and by regularly receiving feedback from his teacher. When the student has matured intellectually and isolates himself for several hours, then (and only then) does he approximate the behavior of a foolproof 1936 automatic machine:

> It will only be when the man is 'concentrating' with a view to eliminating these stimuli or 'distractions' that he approximates a machine without interference.[43]

Turing's "machine intelligence" thus referred to machines that continually interact with their environment during a learning process of sufficiently long duration. Initially, the machinery at hand can be compared to a newborn whose brain is undeveloped:

> All of this suggests that the cortex of the infant is an unorganized machine, which can be organized by suitable interfering training. The organizing might result in the modification of the machine into a universal machine or something like it.[44]

For example, all readers of the present article are "interference machines" (option B) and not automatic machines (option A). However, as soon as one of my readers stops interacting with her peers, she becomes an automatic machine and, possibly, a universal automatic machine that is intelligent in certain areas of discourse. Every

[40] Both Eddington on page 34 in Eddington (1929b) and Turing in Turing (1948) discussed the topic of making mistakes in arithmetic and why, according to them, such mistakes could not be explained away by solely relying on deterministic laws.

[41] A related but not per se compatible narrative appears in Piccinini (2003); cf. Piccinini's observation on page 23 that Turing thought it "possible to reproduce human intelligence … by developing the appropriate kind of digital computer.".

[42] Quoted from pages 421 and 419 in Turing (1948). This source is Turing's draft transcript from 1948; see page 409 in Copeland (2004).

[43] Quoted from page 421 in Turing (1948).

[44] Quoted from page 424 in Turing (1948).

automatic machine has intrinsic limitations, for that is what the results of Gödel, Church and Turing from the 1930s tell us. However, as long as the reader continues to collaborate with fellow readers, engage in discussion with her peers, etc., she will be intrinsically worth more than any fixed universal machine.

To implement the ACE as an "interference machine," Turing programmed it to have one or more "undetermined" states. For any undetermined state, two different follow-up computations were possible. In such cases, the subsequent computation of the ACE was determined, pace Turing's Eddingtonian outlook, by the randomness inherent in nature:

> When a configuration is reached for which the action is undetermined, a random choice for the missing data is made and the appropriate entry is made in the description, tentatively, and is applied.[45]

Turing provided the "random" values as separate inputs to the ACE. (I presume he did this with multiple throws of a die.)

In sum, Turing implemented a learning process in which certain values (of program variables) were first "uncertain," then became "tentative" and then either became "definite" or "uncertain" again, depending on the human-ACE interaction. Turing's learning process relied on "random choices" from an external operator and was consistent with his B-notion of computability from 1936.

Turing's preference for B-computability is at odds with what computer science students are taught today, namely that humans and computers (including yet-to-be-invented machines) will never be able to compute more functions than those that a universal Turing machine can compute. Examples of such grand statements will follow later. While the term "universal Turing machine" covers everything in computer science, this term was far too restrictive for Turing himself. Thus, he wrote in 1948:

> A man provided with paper, pencil, and rubber, and *subject to strict discipline*, is in effect a universal machine.[46]

The emphasized words refer to limitations set out by the Hilbert Program before World War II. However, for Turing, an intelligent human being, such as the creative Gauss, was anything but disciplined, anything but a universal automatic machine.[47]

Turing's 1936 universal machine did, however, come to play an increasingly significant role in the establishment of "computer science" as a new scientific discipline in the USA.

[45] Quoted from page 425 in Turing (1948).

[46] Quoted, with my emphasis, from page 416 in Turing (1948).

[47] Although I do not discuss Turing's imitation game (Turing, 1950) here, my historical interpretation seems to be compliant with Kugel (2002). However, see Gonçalves (2023) for a different, more comprehensive, investigation.

5 Turing's Legacy in the USA

With the end of World War II came the dawn of the computer industry. In the 1940s and early 1950s, specialists in this growing industry converted their mathematically modeled problems into machine notation, which would then serve as input for their computer. Towards the end of the 1950s, some of these specialists were instructing their machine to carry out the translation for them, i.e., from a "program" in a "programming language" into the machine's own code, whence the words "Automatic Programming of Digital Computers" in the quoted passage below.

Researchers in automatic programming sought "machine-independent languages" that were independent of any computer manufacturer (such as IBM). It is precisely in this context that the universal Turing machine began to play a key role in the history of computing. Turing's theoretical concept from 1936 helped a select group of American and British computer programmers to see the wood for the trees. Each tree represented a programming notation (such as FORTRAN and AUTOCODE) and the forest itself corresponded to the brand new, yet-to-be-explored territory of automatic programming (Daylight, 2015). For example, one of those involved, Andrew Booth (1918–2009), spoke the following words in 1959:

> It was Turing who first enunciated the fundamental theorem upon which all studies of automatic programming are based. In its original form the theorem was so buried in a mass of mathematical logic that most readers would find it impossible to see the wood for the trees. Simply enunciated, however, it states that *any computing machine which has the minimum proper number of instructions can simulate any other computing machine, however large the instruction repertoire of the latter.* All forms of automatic programming are merely embodiments of this rather simple theorem and, although from time to time we may be in some doubt as to how FORTRAN, for example, differs from MATHMATIC or the Ferranti AUTOCODE from FLOW-MATIC, it will perhaps make things rather easier to bear in mind that they are simple consequences of Turing's theorem.[48]

The italicized text expresses a fundamental insight of contemporary computer science. All kinds of developed programming notations are essentially equivalent to one other. At best, they can match the mathematical computing power of a universal Turing machine, but never exceed it. In later years, computer scientists have further generalized this statement to yet-to-be-invented programming languages and computers, as the following words by David Harel from 1992 suggest:

> [A]ny algorithmic problem for which we can find an algorithm that can be programmed in *some* programming language, *any* language, running on *some* computer, *any* computer, *even* one that has not been built *yet* but *can be built* … is also solvable by a Turing machine.[49]

[48] Quoted, with my emphasis, from page 1 in Goodman (1960).

[49] Quoted, with my emphasis, from page 233 in Harel (1992).

The previous quotations by Booth and Harel shed light on how Russell's pre-war logicism became algorithmic throughout the second half of the twentieth century, with "algorithmic" referring to the universal Turing machine. Computer scientists (including Harel) take it for granted that the universal Turing machine covers the full load of computability. They regard the universal Turing machine as the most suitable model for all kinds of products designed by engineers (such as iPhones, laptops, desktops, etc.). Engineers, on the other hand, are trained to deal with mathematical models in a completely different way. They rely on a multitude of models for each engineered product and each model they employ has both advantages and disadvantages compared to any other model they use (Daylight, 2016, 2021).

Turing had gained recognition in the USA by the late 1950s, after his death in 1954. Initially, he was regarded by Booth and like-minded specialists as the intellectual father of automatic programming. A few decades later, he was even proclaimed to be the inventor of the modern computer (Bullynck et al., 2015). Over the past 10 years, historians have come to query this claim (Corry, 2017; Daylight, 2012; Mounier-Kuhn, 2012; Price, 2021), much in line with the sequel to Booth's 1959 introduction:

> Why was it, then, that Turing's original work, finished in 1937 before any computing machine of modern type was available, assumed importance only some years after machines were in common use? The reasons, I think, stem entirely from the historical development of the subject.[50]

According to Booth, the very first "computing machines" from the 1940s and early 1950s were "almost exclusively [used] by their constructors"[51] and thus by users who did not abstract from their machine:

> … and, hence, by people who were intimately aware of their internal construction. It took some years before the machines were used for scientific applications, devised by people who were and wanted to remain ignorant of the machine itself and, hence, had to rely on automatic programming techniques.[52]

The power of the universal Turing machine lay, pace Booth, in its abstraction, which only became relevant once applied mathematicians, who were not computer builders, began to register en masse as users of the new technology.

In addition to Turing's posthumous recognition among a select but influential group of American and British computer programmers, I mention in passing that Turing himself came to realize much earlier that one general-purpose machine, such as the ACE, was sufficient to perform the tasks of several special-purpose machines. Andrew Hodges asks whether Turing was the first person ever to appreciate the programmable nature of modern machines in this manner. Not so, as it turns out.[53] A century earlier, Charles Babbage (1791–1871) had already had a penetrating view

[50] Quoted from page 2 in Goodman (1960).
[51] Quoted from page 2 in Goodman (1960).
[52] Quoted from page 2 in Goodman (1960).
[53] See pages 297–298 in Hodges (1983).

on essentially the same matter (Daylight, 2014). Yet neither the latter's work, nor Turing's theoretical work, fundamentally influenced the construction of a first generation of post-war computers.[54] Moreover, many computing professionals came to Turing's insight in a practical way, not via modern logic.[55]

6 Artificial Intelligence

Turing's 1936 article helped establish computer science as an academic discipline in the USA. Only a few years after Booth's speech did American universities begin to offer "computer science" curricula, based on the theoretical Turing machine (Daylight, 2015). In 1966, the first "Turing Prize" was awarded to Alan Perlis (1922–1990), a specialist in automatic programming. Yet the name "Alan Turing" only gained popularity gradually through the last quarter of the twentieth century, if not later.[56]

There is no indication that Booth and Perlis read Turing's 1936 article in full, let alone understood it according to Turing's original line of thought. Rather, their insights came from recast analyses of the early 1950s, written by former doctoral students of Alonzo Church. However, that literature, too, was only partially intelligible. The computer programmer was not a logician; conversely, the average logician knew very little about computers.

Besides Booth and Perlis, John Carr (1923–1997) appropriated modern Turing machinery as a computer programmer. Various computing concepts, such as "simulation," acquired a theoretical underpinning. In Carr's words from 1959:

> If one universal machine can simulate any other machine of a somewhat smaller storage capacity (which is what Turing's statement on universal machines means), it should therefore be possible for a computer to simulate a version of itself with a smaller amount of storage.[57]

With the germination of academic computer science came algorithmic thinking. Every physical phenomenon could, pace Carr, be grasped symbolically via the underlying Turing machine. In Carr's words:

> Based on Turing's proof about universal machines:
>
> 1. Living organisms can be abstractly defined as [a] symbol manipulator.
> 2. Actions of living beings can be described by a program.
> 3. Digital computers have all the features of Universal Turing Machines.
> 4. Digital computers can duplicate human beings[58]

[54] See also page 20 in Haigh and Ceruzzi (2021).
[55] See Chapter 8 in Daylight (2012).
[56] According to, e.g., Peter Naur (Daylight, 2011) and historians (Bullynck et al., 2015).
[57] Quoted from Carr (1959).
[58] Quoted from Carr (1958).

These powerful words, coming from the President of the Association of Computing Machinery, helped build up a support base for American artificial intelligence. The world was algorithmically controllable, including the natural and artificial beings who lived or would come to live in it.

Carr's view on artificial intelligence is consistent with Turing's A-notion of computability and Russell's imaginary bridge, linking the world of physical processes to a logical space. Note, however, that Russell's logical space now consisted solely of computer programs. Fast forward to the present and we see Carr's A-notion of artificial intelligence reigning, as the following words by theoretical computer scientist, Scott Aaronson, vividly illustrate:

> I was lazily relying on the fact that everyone in the room already agreed with me—that ... it was simply self-evident that the human brain is nothing other than a **"hot, wet Turing machine,"** and weird that I would even waste … time with such a **settled** question. Since then, I *think* I've come to a better appreciation of the immense difficulty of these issues—and in particular, of the need to offer arguments that engage people with different philosophical starting-points than one's own.[59]

The last sentence presumably refers to scientists in established disciplines, such as physics, biology and geology. In these sciences, the prevailing idea is that nature contains randomness, which is something that a "hot, wet Turing machine" cannot capture (by its very design). Despite Aaronson's seemingly more cautious stance in the second part of the previous passage, his public appearances confirm that he is a die-hard computer scientist: in principle, any physical process can be grasped intellectually with a computer program (or a Turing machine).[60]

Yet, with mainstream thinking and discipline building comes freethinking. There are plenty of historical players who have come to challenge the Carr-Aaronson establishment, including Carl Hewitt and Giuseppe Longo, whose work I will briefly discuss below. Other freethinkers, whose writings will not be reviewed here, include Carl Petri, Peter Wegner, Dina Goldin, Jan van Leeuwen, B. Jack Copeland, Oron Shagrir, Mark Burgin, Graham Priest and Edward A. Lee.

7 Freethinkers

In the 1970s, Carl Hewitt began to take issue with the universal Turing machine as an overarching theoretical concept. Contrary to the classical, functional view of computability, he described "computation" as a "cooperating society of 'little men' each of whom can address others with whom it is acquainted and politely request that some task be performed."[61] Hewitt viewed the real world, including the world of interrupts, messages (emails) and computer networks, as intrinsically indeterminate.

[59] Quoted from page xxviii in Aaronson (2013), with Aaronson's italics and my boldface.
[60] See Aaronson's Bernays lectures (Aaronson, 2019).
[61] Quoted from page 236 in Hewitt et al. (1973).

The following words from his former doctoral student, Gul Agha, nicely sum up Hewitt's B-view:

> In any real network of computational agents, one cannot predict precisely when a communication sent by one agent will arrive at another. ... Therefore a realistic model must assume that the arrival order of communications sent … is *physically indeterminate*.[62]

Hewitt's modeling language does not abstract away the "physically indeterminate" character which engineers of distributed systems consider in their work. From Hewitt's perspective, Russell's logical atomism was futile:

> The actor paradigm [of Carl Hewitt] stands in sharp contrast to the logicist approach, which not only postulates the existence of a unique reality, but commits us to representing our knowledge in terms of a consistent collection of information (Agha, 1987).

Hewitt's postmodernist attitude to symbolic logic is largely unexplored territory in present-day computer science.[63] The gist is that while Hewitt's actor paradigm does not guarantee algorithmic control in the Carr-Aaronson spirit, it has nonetheless served the software industry well.[64]

Theoretical computer scientist Longo also abandoned the standard view. In 1995, he wrote the reverse of what was posited above by Aaronson:

> Nobody seems to doubt that our brain is a massively parallel, distributed, interactive device, even though a few still try to reduce it to Turing machines and claim that, "in principle", any finite piece of the world should be fully describable by symbolic manipulations.[65]

The last words ("any finite piece of …") summarize the most extreme view of members of Camp B, although Longo initially belonged to Carr and Aaronson's Camp A.[66] To express his dismay, Longo recently wrote a letter to the late Turing, denouncing the notion that "everything is computational" and querying "the myth of the universe as a Turing Machine, against your very precise observations," referring to Turing's insights, which Longo shares.[67] Subsequently, Longo pointed to his colleagues in computer science "who are using the only technique that they know … flattening it onto a universe … made only of formal calculations," and expressed

[62] Quoted, with my emphasis, from page 15 in Agha (1986).

[63] Promising work in this regard comes, e.g., from Priest (2008).

[64] See the recollections provided in Hoare et al. (2019).

[65] See Sect. 7 in Longo (1997).

[66] On page 85 in his book, Aaronson opens his intellectual door a crack by allowing his "hot, wet Turing machine" to appeal to a polynomial amount of non-Turing-computable information (Aaronson, 2013). In this way Aaronson complies slightly more with Darwin's theory of evolution, which, as far as Longo is concerned, is anything but Turing-computable. It is up to Aaronson to further clarify his Weltanschauung, though he remains miles apart from Longo's standpoint.

[67] See page 89 in Longo (2018).

agreement with Turing's position that there is no reason to regard the universal Turing machine as the limit of computability:

> ..... as if yours is the last machine that man will be capable of inventing ... I am convinced that we shall invent others ...[68]

Engineering is, pace Longo, not limited by Turing's 1936 theory.

In a similar vein, Andrew Hodges has recently suggested that the human brain was anything but a hot, wet Turing machine for Turing (Copeland et al., 2017). Specifically, Hodges wrote in 2012:

> [Turing] was also one of the first to use a computer for simulating physical systems. In 1951, however, Turing gave a radio talk with a different take on this question, suggesting that the nature of quantum mechanics might make simulation of the physical brain impossible.[69]

In this article, I have proposed to interpret Turing as a member of Camp B from the very start of his university studies. According to Eddington, Turing, Hewitt, Longo and other B-members, the gap between brain processes and symbolic logic is difficult, if not impossible, to bridge. It is definitely not feasible if one can only resort to Turing machinery.

8 Conclusions

For Turing, his 1936 impossibility result did not, in general, apply to human mathematicians or to actual, programmable devices. His automatic machines had only served to formally capture the notion of a disciplined human computer, in line with Russellian and Hilbertian intellectual developments in the first third of the twentieth century, and to reveal their intrinsic limitations. After the war, Turing wanted to use the ACE machine so that it would resemble a creative mathematician rather than his automatic machinery from 1936. Hence, he turned to Eddington's physical indeterminism—i.e., to "random" inputs for the ACE—to provide room for the making of errors, akin to those made by a child that learns.

Despite Turing's preference for his broader B-notion of computability, computer scientists today follow a neo-Russellian tenet; that is, they look at actual computing devices through algorithmic glasses, in compliance with Turing's A-notion of computability.[70] While computer science takes Turing's universal machine as the limit of all achievable forms of computability, it was explicitly perceived as banal by Turing in 1948. From his perspective, a creative person is significantly more than that which a fixed symbolic logic or a universal Turing machine can offer. Human creativity cannot be fully captured by a series of logical rules, that is, by a program text written in a programming language. Conceptually, Turing's machine intelligence thus

[68] Quoted from page 89 in Longo (2018).

[69] Quoted from page 164 in Hodges (2012).

[70] For a follow-up on this line of research, see Daylight and Schüttpelz (2022).

differed significantly from American artificial intelligence, with which many readers are more familiar.

In the decades following his death in 1954, Turing's universal machine has become the quintessential model of the modern computer and, by extension, of every process that physics has to offer. In my present contribution I provide reasons to suggest that Turing would have challenged the dictum that the computing power of any yet-to-be-invented machine is, a priori, limited by that of a universal Turing machine. In contrast to Turing, computer scientists generally regard the universal Turing machine as the most suitable model for all kinds of engineered products (e.g., iPhones, laptops, desktops) presumably because program productivity hinges on a digital (Turing machine) abstraction of physical reality. All programming languages are disguised universal Turing machines, according to the theoretical computer scientist. In retrospect, then, it is unsurprising that computer science portrays Turing as the intellectual father, and even as the inventor, of the modern computer.

A detailed chronology pertaining to several episodes in the intellectual life of the true Turing is forthcoming. This article already provides evidence to support the claim that Turing, himself, did not view computations in the real world to be exhaustively characterized by his automatic machines from 1936. The Turing machine is dead. Long live Turing!

Acknowledgements The author, also known as Karel Van Oudheusden, was financed by *Sonderforschungsbereiche* SFB 1187 "Medien der Kooperation" (Siegen University) and by *Agence nationale de la recherche* ANR-17-CE38-003-01 "PROGRAMme" (Lille University). He is grateful for any feedback on his research from Erhard Schüttpelz, Bernardo Gonçalves, Julian Rohrhuber, Philip Dutré, Leo Corry, Edward A. Lee, David Nofre and two anonymous reviewers, in particular. He also thanks the editors, Juan Luis Gastaldi and Luc Pellissier (guest editors) and Mariarosaria Taddeo (editor-in-chief). Finally, he dedicates this article to Bart Demoen. An online lecture on the contents of this article is available at: https://dijkstrascry.com/lecture3

Funding Open Access funding enabled and organized by Projekt DEAL.

Declarations

Conflict of interest Part of this article is based on a translation of the present author's contribution in Van Oudheusden and Dutré (2022) which comprises lecture notes for students at KU Leuven. The author has maintained all rights in this regard, including copyright. The author has nothing further to declare.

References

Aaronson, S. (2013). *Quantum computing since democritus*. Cambridge University Press.

Aaronson, S. (2019). *Quantum Computing and the Fundamental Limits of Computation.* Paul Bernays Lectures 2019—ETH Zurich. Retrieved February 10, 2022, from https://gess.ethz.ch/en/news-and-events/paul-bernays-lectures/bernays-2019.html. Accessed 10 2022

Agha, G. (1986). *Actors: A Model of Concurrent Computation in Distributed Systems*. MIT Press.

Agha, G. (1987). Concurrent programming using actors. In A. Yonezawa & M. Tokoro (Eds.), *Object-Oriented Concurrent Programming* (pp. 37–53). MIT Press.

Bullynck, M., Daylight, E. G., & De Mol, L. (2015). Why did computer science make a hero out of Turing? *Communications of the ACM, 58*(3), 37–39.

Carr, J. W. (1958). Languages, logic, learning, and computers. *Computers and Automation, 7*(21–22), 25–26.

Carr, J. W., et al. (1959). Programming and coding. In E. Grabbe (Ed.), *Handbook of Automation, Computation, and Control: Computers and Data Processing (Chapter 2).* Wiley.

Church, A. (1937). Review: A.M. Turing, on computable numbers, with an application to the *Entscheidungsproblem*. *Journal of Symbolic Logic, 2*(1), 42–43.

Copeland, B. J. (Ed.). (2004). *The Essential Turing.* Clarendon Press.

Copeland, B. J., Sprevak, M., & Shagrir, O., et al. (2017). Is the whole universe a computer? In B. J. Copeland (Ed.), *The Turing Guide: Life, Work, Legacy* (pp. 445–462). Oxford University Press.

Corry, L. (2008). The development of the idea of proof. In T. Gowers, J. Barrow-Green, & I. Leader (Eds.), *Princeton Companion to Mathematics* (pp. 129–142). Princeton University Press.

Corry, L. (2015). *A Brief History of Numbers*. Oxford University Press.

Corry, L. (2017). Turing's Pre-War Analog Computers: The Fatherhood of the Modern Computer Revisited. *Communications of the ACM, 60*(8), 50–58.

Davis, M. (2000). *Engines of Logic, Mathematicians and the Origins of the Computer*. W.W. Norton & Company.

Daylight, E. G. (2011). *Pluralism in software engineering: Turing award winner peter naur explains*. Lonely Scholar.

Daylight, E. G. (2012). *The Dawn of Software Engineering: from Turing to Dijkstra*. Lonely Scholar.

Daylight, E. G. (2014). A Turing Tale. *Communications of the ACM, 57*(10), 36–38.

Daylight, E. G. (2015). Towards a Historical Notion of 'Turing—the Father of Computer Science.' *History and Philosophy of Logic, 36*(3), 205–228.

Daylight, E. G. (2016). *Turing Tales*. Lonely Scholar.

Daylight, E. G. (2021). Addressing the Question "What is a Program Text?" via Turing Scholarship. *IEEE Annals of the History of Computing, 43*(4), 87–91.

Daylight, E. G., Schüttpelz, E. (2022). "The Turing Machine as a Boundary Object: Sorting out American Science and European Engineering." Filmed July 2022 at WAI22: Wittgenstein and AI, London. Video, https://www.dijkstrascry.com/Daylight_Schuettpelz_BoundaryObject.

Dingle, H. (1954). *The Sources of Eddington's Philosophy*. Cambridge University Press.

Eddington, A. S. (1920). The Internal Constitution of the Stars. *The Scientific Monthly, 11*(4), 297–303.

Eddington, A. S. (1929a). *The Nature of the Physical World*. The Macmillan Company.

Eddington, A. S. (1929b). *Science and the Unseen World: Swarthmore Lecture*. Quaker books.

Fan, Z. (2020). Hobson's Conception of Definable Numbers. *History and Philosophy of Logic, 41*(2), 128–139.

Gonçalves, B. (2023). The Turing Test is a Thought Experiment. *Minds and Machines, 33*, 1–31. https://doi.org/10.1007/s11023-022-09616-8

Goodman, R. (Ed.) (1960). *Annual Review in Automatic Programming I: Papers read at the Working Conference on Automatic Programming of Digital Computers held at Brighton, 1–3.* Pergamon Press.

Haigh, T., & Ceruzzi, P. E. (2021). *A New History of Modern Computing*. MIT Press.

Harel, D. (1992). *Algorithms: The Spirit of Computing*. Addison Wesley.

Hewitt, C., Bishop, P., Steiger, R. (1973). A Universal Modular ACTOR Formalism for Artificial Intelligence. In W. Kaufmann (Ed.), *Proceedings of the 3rd International Joint Conference on Artificial Intelligence* (pp. 235–245). Stanford.

Hoare, C. A. R., Armstrong, J., Hewitt, C. (2019). *Let's #TalkConcurrency: "Panel Discussion with Sir Tony Hoare, Joe Armstrong, and Carl Hewitt."* YouTube video with transcript (February 19, 2019). Retrieved February 10, 2022, from https://www.erlang-solutions.com/blog/lets-talkconcurrency-panel-discussion-with-sir-tony-hoare-joe-armstrong-and-carl-hewitt/

Hobson, E. W. (1910). *Mathematical and Physical Science.* Address to the British Association in 1910. Retrieved February 10, 2022, from https://mathshistory.st-andrews.ac.uk/Extras/BA_1910_1/

Hobson, E. W. (1921). *The Theory of Functions of a Real Variable and The Theory of Fourier's Series V1*. Cambridge at the University Press, (2nd edition).
Hodges, A. (1983). *Alan Turing: The Enigma*. Burnett Books.
Hodges, A. (2001). Turing. In R. Monk & F. Raphael (Eds.), *The Great Philosophers: From Socrates to Turing* (pp. 493–541). Phoenix.
Hodges, A. (2012). Beyond Turing's Machines. *Science, 336*(6078), 163–164.
Irvine, A. (2017). *Bertrand Russell*. The Stanford Encyclopedia of Philosophy (Summer 2017 Edition). In E. N. Zalta (Ed.). Retrieved February 10, 2022, from https://plato.stanford.edu/archives/sum2017/entries/russell/
Jeans, J. (1930). *The Mysterious Universe*. Cambridge University Press.
Kugel, P. (2002). Computing Machines Can't Be Intelligent (…and Turing Said So). *Minds and Machines, 12*(4), 563–579.
Longo, G. (1997). The difference between Clocks and Today's Computing Machines. *La Nuova Critica, 29*(1), 31–42.
Longo, G. (2018). Letter to Turing. *Theory, Culture & Society, 36*(6), 73–94.
Mancosu, P. (1999). Between Russell and Hilbert: Behmann on the Foundations of Mathematics. *The Bulletin of Symbolic Logic, 5*(3), 303–330.
Mander, W. J. (2014). *British Idealism: A History*. Oxford University Press.
McDaniel, K. (2020). *John M. E. McTaggart*. The Stanford Encyclopedia of Philosophy (Summer 2020 Edition). In E. N. Zalta (Ed.). Retrieved February 10, 2022, from https://plato.stanford.edu/archives/sum2020/entries/mctaggart/
Millican, P. (2021). Alan Turing and Human-Like Intelligence. In S. Muggleton & N. Chater (Eds.), *Human-Like Machine Intelligence (Chapter 2).* Oxford University Press.
Moore, G. (1982). *Zermelo's Axiom of Choice: Its Origins, Development, & Influence*. Dover Publications Inc.
Mounier-Kuhn, P. (2012). Logic and Computing in France: A Late Convergence. In L. De Mol & G. Primiero (Eds.), *AISB/IACAP World Congress 2012—History and Philosophy of Programming* (pp. 44–47).
Piccinini, G. (2003). Alan Turing and the Mathematical Objection. *Minds and Machines, 13*(1), 23–48.
Price, D. (2021). *Geniuses at War: Bletchley Park, Colossus, and the Dawn of the Digital Age*. Knopf.
Priest, G. (2008). *An Introduction to Non-Classical Logic*. Cambridge University Press.
Rochelle, G. (2018). *Behind Time: The Incoherence of Time and McTaggart's Atemporal Replacement*. Routledge.
Russell, B. (2009). *The Principles of Mathematics*. Routledge.
Schmidt, K. (2011). Dispelling the Mythology of Computational Artifacts. In K. Schmidt (Ed.), *Cooperative Work and Coordinative Practices* (pp. 391–413). Springer-Verlag.
Shanker, S. (1987). Wittgenstein versus Turing on the Nature of Church's Thesis. *Notre Dame Journal of Formal Logic, 28*(4), 615–649.
Sieg, W. (2013). *Hilbert's programs and beyond*. Oxford University Press.
Turing, A. M. (1936-7). On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society, s2–42*(1), 230–265.
Turing, A. M. (1938). Systems of Logic Based on Ordinals. PhD dissertation, Princeton University. Reprinted in 2004. In B. Jack Copeland (Ed.), *The Essential Turing* (pp. 146–204). Clarendon Press.
Turing, A. M. (1948). Intelligent Machinery, 1948 report for National Physical Laboratory. Reprinted in 2004. In B. Jack Copeland (Ed.), *The Essential Turing* (pp. 410–432). Clarendon Press.
Turing, A. M. (1950). Computing machinery and intelligence. *Mind, LIX*(236), 433–460.
van Dongen, J. (2010). *Einstein's Unification*. Cambridge University Press.
Van Oudheusden, K., & Dutré, P. (2022). Alan Turing: Van 'Wat is berekenbaarheid?' tot 'Wat is artificiële intelligentie?' In P. d'Hoine & B. Pattyn (Eds.), *Denken over onze oorsprong: Lessen voor de eenentwintigste eeuw.* Universitaire Pers Leuven.
Vibert, D. (2021). *Arthur Eddington*. Encyclopedia Britannica. Retrieved February 10, 2022, from https://www.britannica.com/biography/Arthur-Eddington