Early AI in Britain: Turing et al.

B. Jack Copeland, *University of Canterbury, Christchurch, 8041, New Zealand*

This article presents an overview of Turing's early contributions to machine intelligence, together with a summary of his influence on other early practitioners. Following his famous work on the Entscheidungsproblem in the 1930s, Turing staked out the field of machine intelligence during the 1940s. His wartime Bombe used what we now call heuristic search to do work requiring intelligence when done by humans. In key papers in 1948 and 1950 he discussed search, learning, robotics, chess, the theorem-proving approach to AI, the genetic algorithm concept, and artificial neural networks, as well as introducing the Turing test. He influenced the first generation of programmers in Britain, whose pioneering contributions to machine intelligence included work on board games, learning, language-processing, and reasoning—contributions made years before the term "Artificial Intelligence" was coined at Dartmouth in 1956.

CONTEXT: THINKING MACHINES, DARTMOUTH, THE TURING MACHINE, AND THE ENTSCHEIDUNGSPROBLEM

Tracing far back along the chain of intellectual precursors to modern AI, one reaches the American philosopher C. S. Peirce. In 1908, Peirce expressed the idea that a machine—"some Babbage's analytical engine or some logical machine"—is capable of all mathematical reasoning [47, p. 434]. He was skeptical, however, calling the idea "malignant," and firmly placing it among others he deemed "logical heresies" (ibid.).

Peirce was a powerful influence on Hilbert and his group at Göttingen [27, p. 1]. In 1903, Peirce formulated the *decision problem for first-order logic* in roughly the form in which Turing later tackled it [46]. At Göttingen, the decision problem was named the *Entscheidungsproblem*, it seems by Behmann, a young member of Hilbert's group [34]. As early as 1921, Behmann used the concept of a *machine* to clarify the nature of the Entscheidungsproblem, saying: "One might, if one wanted to, speak of mechanical or machine-like thinking" and "Perhaps one can one day even let it be carried out by a machine" [34, p. 176].

From Göttingen, the Entscheidungsproblem travelled to Cambridge and Turing. His encounter with the Entscheidungsproblem launched him on the trajectory that led to his theoretical work on what he called "logical computing machines" and then, eventually, to his practical work on the hardware and software designs for the ACE and other early British electronic computers [9], [18], [29], [30]. This same trajectory issued in his early work on what he called "intelligent machinery." Researchers that he influenced termed the new field *machine intelligence*.

The later term "Artificial Intelligence" appeared at a 1956 conference in the U.S., the Dartmouth Summer Research Project on Artificial Intelligence, organized by McCarthy and Shannon. (The term is usually credited to McCarthy, but he said in an interview: "I won't *swear* that I hadn't seen it before . . . Someone may have used it in a paper or a conversation" [36, p. 96].) It appears that McCarthy knew little or nothing at this time about preexisting British work on machine intelligence, nor about computer developments in Britain in general—even saying in *Scientific American* in 1966: "it does not seem that the work of . . . Turing . . . played any direct role in the labors of the men who made the computer a reality" [35, p. 68].

This century has certainly seen a radical reappraisal of Turing's role in the development of computing, and he is now widely regarded as a "founding father" of computer science (although some question the accuracy of this reappraisal, e.g., Vardi [72]). The time may be ripe for the AI community to reappraise Turing's position vis-à-vis the field of AI and its origins.

Digital Object Identifier 10.1109/MAHC.2023.3300660
Date of publication 1 August 2023; date of current version 1 September 2023.

This article does not itself attempt such a reappraisal, however, the aim being simply to give an overview of Turing's pioneering work in machine intelligence—which is still not as widely known as it might be. The article therefore traces the early developments involving Turing and researchers he influenced.

Turing's first significant brush with the Entscheidungsproblem was—so far as we know—in a 1935 lecture by the Cambridge logician Newman, himself soon to become a presence in debates on machine intelligence. Newman said in an interview, "I believe it all started because [Turing] attended a lecture of mine on foundations of mathematics and logic."[1]

Turing proved that the Entscheidungsproblem is *un*solvable, by means of an ingenious "diagonal" argument.[2] A step along the way was his invention of what he called the "universal computing machine"—now simply the universal Turing machine and widely regarded as a bare-bones logical model of almost every modern electronic digital computer.[3] This invention in turn led to Turing's pioneering investigations into machine intelligence.

In his classic 1936 paper on the Entscheidungsproblem [61], there was no mention of machine intelligence per se, but central to the paper's argument was a crucial thesis, now named "Turing's thesis" (and also the "Church-Turing thesis"):

> The "computable" numbers include all numbers which would naturally be regarded as computable [61, p. 74].

TURING'S THESIS AND THE MECHANIZATION OF INTELLIGENCE

This thesis, for which Turing argued strenuously, states that a Turing machine can in principle do any of the mathematics that human clerks—human computers—are capable of doing. In Turing's pithy formulation, the "computable" numbers are the numbers produced by the Turing machine, and the numbers "which would naturally be regarded as computable" are numbers producible by human computers.

The work of human computers—there were large numbers of them in Turing's day—would normally be said to have required a form of intelligence, even though the work needed no ingenuity on the part of the human clerk nor any mathematical "intuition." A glance at Skan's classic *Handbook for Computers* [57] shows the degree of intelligence that the work of human computers demanded. Junior computers recruited by the National Physical Laboratory (where Turing worked in the immediate postwar years) were typically selected from among school leavers with mathematical qualifications. The training of these young computers included an emphasis on such skills as error-management: "At school," they were told, "if you made an error you were punished by loss of marks. Here, errors will be made all the time. They must not leave the building" [73, p. 266]. A computer's basic proficiencies included finding roots, numerical differentiation and integration of functions, numerical accuracy checks, and the solution of multivariable equations by a wide range of methods [57]. Advanced computational methods would require computers selected from among university graduates [73, p. 265]. Turing's thesis maintained that all work done by human computers can be done by his machines.

If—as Minsky stated in the 1960s, in his well-known characterization of AI [41, p. v]—"Artificial Intelligence is the science of making machines do things that would require intelligence if done by men," then Turing's thesis entails the in-principle achievability of at least a certain level of AI. The thesis implies that suitably programmed Turing machines are able to do work that is naturally regarded as requiring intelligence when done by humans (assuming one accepts that the work of human computers is normally taken to require a certain degree of intelligence).

By 1938, Turing was arguing, further, that what he called "the exercise of ingenuity in mathematics" is reducible to processes that a Turing machine can carry out: "ingenuity is replaced by patience," he argued [62, pp. 192–193]. Intuition, though, which he distinguished from ingenuity, cannot be wholly replaced by patience, he maintained (ibid.). He argued that "finding a formal logic which wholly eliminates the necessity of using intuition" is an "impossibility" [62, p. 193].

There was, to be sure, a deflationary aspect to Turing's later discussions of intelligence:

> The extent to which we regard something as behaving in an intelligent manner is determined as much by our own state of mind and training as by the properties of the object under

[1]Newman interviewed by Evans (circa 1977), "The Pioneers of Computing: An Oral History of Computing," London, U.K.: Science Museum; also relevant are Smithies' notes taken in the course the previous year (F. Smithies, "Foundations of Mathematics. Mr. Newman," 1934, St John's College Library, Cambridge, GB 275 Smithies/H/H57).

[2]For further information on the Entscheidungsproblem and Turing's attack on it, as well as Church's independent contributions, see [14].

[3]What Turing called "computing machines" were dubbed "Turing machines" by Church in his review of Turing's paper [11, p. 43].

> consideration. If we are able to explain or predict its behaviour . . . we have little temptation to imagine intelligence. With the same object therefore it is possible that one man would consider it as intelligent and another would not; the second man would have found out the rules of its behaviour [66, p. 431].

Nevertheless, looking back at Turing's two classic papers of the 1930s [61], [62], we can fairly say that, first, his work led to the anchoring of discussions of thinking machines (previously lacking in detail, or even fanciful) to a definite and powerful machine architecture, the Turing machine; and that, second, he established (by means of a bouquet of abstract arguments) a thesis implying that Turing machines are able to perform tasks commonly said to require a form of intelligence when done by humans.

These achievements alone might be sufficient ground to say that Turing's work occupies a foundational position in the history of the mechanization of intelligence, but he in fact contributed much more. He argued forcefully that foreseeable electronic computers are capable of "showing intelligence"—especially if the computer has the ability to learn, in which case it would, he said, ultimately "be like a pupil who had learnt much from his master, but had added much more by his own work" [65, p. 393]. "What we want is a machine that can learn from experience," he said, and "the machine must be allowed to have contact with human beings in order that it may adapt itself to their standards" [65, pp. 393–394].

These farsighted statements were made in an address he gave in 1947 to the London Mathematical Society—seemingly the first time the idea of computer intelligence was aired in public in a lecture by an actual practitioner in the emerging field. For by that time, Turing was indeed a practitioner. Following the publication of his two classic papers of the 1930s, he had begun to explore the potential for the practical development of machines showing intelligence, ingenuity, originality, the ability to learn, and more.

BLETCHLEY PARK: THE POWER OF SEARCH

In the 1950s, two leading pioneers of AI in the U.S., Newell and Simon, emphasized the importance of guided search. They used the term "heuristic search," adding the noun "heuristic"[4] to the vocabulary of computer science in an influential 1957 paper:

> A process that *may* solve a given problem, but offers no guarantees of doing so, is called a *heuristic* [43, p. 220].

A heuristic is a mechanical process that—in terms of achieving its goal—is "fallible but 'fairly reliable'" [25, pp. 83-84]. A good heuristic works often enough to be useful.

In their joint Turing Award lecture in 1975, Newell and Simon summarized the approach their AI research had followed since the mid 1950s. The fundamental idea was that a (symbol-processing) "system exercises its intelligence in problem solving by search"; and they defined search as "generating potential solutions and testing them" [44, pp. 120, 126].[5] A physical system, they said, "must use heuristic search to solve problems because such systems have limited processing resources" [44, p. 120]. Hence, their *Heuristic Search Hypothesis*:

> [S]ystems solve problems by using the processes of heuristic search. (ibid.)

They described this as a "law of qualitative structure for AI" [44, p. 126].

In their Turing Award lecture, Newell and Simon spoke from time to time of Turing machines but—ironically—there was no mention of Turing's work in machine intelligence. Newell and Simon of course knew nothing about what Turing and his fellow codebreakers had done at Bletchley Park during the Second World War, since the relevant documents were not declassified until 20 or more years after their lecture. Heuristic search (although not by that name) was in fact a principal weapon in the codebreakers' armory.

Moving to Bletchley Park the day after Prime Minister Chamberlain declared war on Hitler, Turing worked on Enigma and (with his colleague Welchman) designed the Bombe, an electromechanical behemoth for attacking encrypted German messages. The relay-based Bombes (each containing around one million soldered connections [33, p. 291]) turned Bletchley Park into a codebreaking factory: By late 1943, these engines of search were achieving a total average throughput of 84,000 broken messages each month—two messages every minute, 24/7 [28, p. 29].

The Bombe trawled at superhuman speed through the Enigma machine's possible settings, looking for the ones the machine's operator had used to encrypt the message. Because of the astronomical number of potential settings, an *exhaustive* search was out of

[4]For some context, see [49].

[5]The systems they considered were what they called "symbol systems," exemplified by Turing machines.

the question; searches had to be guided into promising regions of the solution space.

One way of guiding the search involved the use of what Turing called "multiple encipherments" [63, p. 317]. These were properties of a *crib*, a word or phrase (such as WETTER FUER DIE NACHT) that the codebreaker thought might be part of the concealed German message. Each letter of a crib is enciphered by the Enigma machine as some other letter (*X* as *Y*, say). A multiple encipherment, or loop, occurs if there is then, further on in the coded version of the message, another occurrence of *Y*, and this deciphers to *X* in the crib. Such loops do not happen very often, because it is much more likely that the second *Y* would decipher to some other letter. However, a good crib needs to contain a longer loop, such as *A* enciphering to *B* and then a later occurrence of *B* deciphering to *C* and then, further on still, *C* enciphering to *A*, and a crib could contain a loop involving four or more letters. These multiple encipherments were the basic tool for guiding the Bombe's search.

A search guided by a multiple encipherment might fail. In Newell and Simon's phrase there were "no guarantees," but the heuristics used with the Bombe worked often enough. Another of the various examples of heuristics that Turing described (in his 1940 write-up of the Bombe) was named *Herivelismus*, after its inventor Herivel. Herivelismus relied on the fact that, when German Enigma operators came on duty and enciphered their first message, a proportion tended not to make too thorough a job of turning the Enigma machine's three code-wheels away from the wheels' "base" position (picture the wheels as resembling the wheels of a combination lock). When the Bombe was searching for the positions of the code-wheels at the start of what was believed to be the first message of a shift, only positions in the base position's *neighborhood* were considered [63, p. 335]. Herivel's insight was reduced to practice in the form of a mechanical process for finding the wheel-positions that was fallible yet fairly reliable. This heuristic broke many messages. Before heuristic search even had a name, it played a dramatic role in the mechanization of thought processes.

Not long after the war ended (when the still highly secret Bombes were mothballed) Turing began extolling the use of search for creating what he called "intelligent machinery." He advanced what can be termed *Turing's Search Principle*:

> [I]ntellectual activity consists mainly of various kinds of search [66, p. 431].

An awareness of the connection between intellectual activity and mechanical search was already present in his prewar work, where he linked the activity of logico-mathematical proof with search. (An essential component of this idea, the concept of an enumeration of the provable formulae, is present in Hilbert's 1904 classic [26].) Turing expressed his proof-as-search idea like this:

> We are always able to obtain from the rules of a formal logic a method of enumerating the propositions proved by its means. We then imagine that all proofs take the form of a search through this enumeration for the theorem for which a proof is desired [62, p. 193].

It is in that manner, Turing said, that ingenuity is replaceable by patience. (He actually used the word "heuristic" on this page of his discussion, although not it seems in the Newell-Simon sense (ibid.).) It was not until the Bombes were in operation, however, that Turing saw the ample successes of the practical, large-scale use of fallible mechanical searches for replacing—to an extent—the endeavors of human codebreakers.

As the Newell–Simon approach eventually demonstrated, Turing was quite right when—almost a decade before Newell and Simon presented their first heuristic program at Shannon and McCarthy's Dartmouth conference[6]—he predicted that "research into intelligence of machinery will probably be very greatly concerned with 'searches'", [66, p. 430].

At Bletchley Park, Turing even composed a typescript about machine intelligence (see the next section). Now lost, this was undoubtedly the earliest paper in the field.

CHESS AND MECHANIZED LEARNING

Michie, a leading figure in 20th-century British AI, was one of Turing's younger colleagues at Bletchley Park. When I interviewed him at his home near Palm Springs in 1998, he put on record important glimpses of Turing's wartime thinking. (Of course, recollections of what happened more than half-a-century previously must always be handled with care, especially when the recollector played a key role in later relevant events—with the attendant risk of anachronism.) Some evenings Turing and Michie would retreat to a pub to discuss their favorite topic:

DONALD MICHIE: "What the codebreaker does is very much a set of intellectual operations and thought processes, and so we were thoroughly familiar with the idea of automating thought processes—both of us were up to our elbows in automation of one kind and another."

[6]There are some interesting first-hand accounts of this presentation in [36, pp. 104–108].

Chess, which Michie later described as "the *Drosophila melanogaster* of machine intelligence" [39, pp. 78–79], provided a convenient framework for their theorizing. Good, another codebreaker, would sometimes join in these ongoing wartime discussions, usually during Sunday morning walks with Turing and Michie. He told me of an even earlier conversation with Turing, in 1941, before Michie's time at Bletchley Park, when they had "talked about the possibility of mechanizing chess."[7] I asked Michie what he recalled of his historic discussions with Turing.

MICHIE: "There were three headings. One was: methods of mechanizing the game of chess and games of similar structure. Another was: the possibility of machine algorithms and systems which could learn from experience; and the third area was a little more general, to do with the possibility of instructing machines with more general statements than purely ground-level factual statements—which would involve, in some sense, the machine understanding and drawing inferences from what it was told."

COPELAND: "Did you and Turing often play chess together at this time?"

MICHIE: "Being one of the few people in the Bletchley environment bad enough to give him a reasonably even game, I became his regular sparring partner. Our discussions on machine intelligence started from the moment that we began to play chess together."

COPELAND: "What specific proposals did Turing make concerning chess programming at this time?"

MICHIE: "We talked about putting numerical values on the pieces. And we talked about the mechanization of conditionals of the form: "If I do *that*, he might do *that*, or alternatively he might do *that*, in which case I could do *this*"—what today we would call look-ahead. We certainly discussed priority of ordering according to some measurable plausibility of the move."

COPELAND: "What else?"

MICHIE: "Our discussions of chess programming also included the idea of punishing the machine in some sense for obvious blunders, and the question of how on earth it might be possible to make this useful, beyond the pure rote-learning effects of punishment—deleting a move from the repertoire. Which is only applicable in the early opening.'"

COPELAND: "Did Turing have any specific ideas about how to do that?"

MICHIE: "I don't remember any in the context of chess. I do remember him later circulating a typescript in which he had specific ideas about learning and more general varieties of learning."

COPELAND: "Was this at Bletchley?"

MICHIE: "At Bletchley, yes. When I say 'circulating,' I know he showed a copy to me and to Jack Good—and I suppose to one or two other associates—for our comments."

None of this Bletchley-era pioneering work on computer chess was published. Like Turing and Michie, Good "made the mistake of thinking it was not worth publishing," he said.[8]

When peace descended, Turing had more time to think about machine intelligence. In 1945, the National Physical Laboratory hired him to design an electronic computer, the Automatic Computing Engine or ACE—a universal Turing machine in hardware. (Womersley, who coined the name "Automatic Computing Engine," had visited Aiken's computer at Harvard in 1945, calling it "Turing in hardware" [76].) The ACE project was Turing's opportunity to consider the possibilities of machine intelligence in the exciting new context of electronic, general-purpose, stored-program hardware. He said that his aim was to "make a brain,"[9] and a letter he wrote from the National Physical Laboratory contained some remarkable statements:

> In working on the ACE I am more interested in the possibility of producing models of the action of the brain than in the practical applications to computing . . . The ACE will be used . . . in the first instance in an entirely disciplined manner . . . It will also be necessarily devoid of anything that could be called originality. There is, however, no reason why the machine should always be used in such a manner: there is nothing in its construction which obliges us to do so. It would be quite possible for the machine to try out variations of behaviour and accept or reject them . . . and I have been hoping to make the machine do this.[10]

Turing even mentioned computer intelligence in his otherwise austere report setting out the design of the ACE:

> Given a position in chess the machine could be made to list all the "winning combinations" to a depth of about three moves on either side. This . . . raises the question "Can the machine play chess?" It could fairly easily be made to play a rather bad

[7]I benefitted from interviewing Good in 2004.

[8]Good quoted in [[3], p. 14].
[9]Letter from Bayley to Copeland, 6 May 1998.
[10]Letter from Turing to Ross Ashby, undated, circa 1946, http://www.alanturing.net/turing_ashby.

> game . . . There are indications however that it is possible to make the machine display intelligence at the risk of its making occasional serious mistakes. By following up this aspect the machine could probably be made to play very good chess [64, p. 389].

In the summer of 1948, Turing created a chess-player in the form of what he called a "paper machine," a system of machine-rules in loose pseudocode. The rules were acted out using paper and pencil to do the computations [66, p. 431]. The system that Turing and his collaborator Champernowne called "Turochamp" proved capable of beating a human player, described as a "beginner at chess" [10]. Champernowne summed up:

> Our general conclusion was that a computer should be fairly easy to programme to play a game of chess against a beginner and stand a fair chance of winning or at least reaching a winning position [10].

In 1948, Turing left the National Physical Laboratory for the University of Manchester's new Computing Machine Laboratory, joining Newman.[11] There he continued his work on machine intelligence, telling a newspaper reporter that he saw no reason why electronic computers should not "enter any one of the fields normally covered by the human intellect, and eventually compete on equal terms."[12]

At the Manchester lab, Turing delved further into computer chess, describing the successor to Turochamp in a typescript [70] completed in 1951[13] (it was published in 1953, with modifications, in [5]). Turing had mentioned in the course of his 1947 lecture to the London Mathematical Society that he and Shannon discussed computer chess [65, p. 393]. Their individual ground-breaking papers on the subject—Shannon's published in 1950 but written in 1948, a few months after Turing's Turochamp beat a human player [55]—set out the state of the art in the new field. Using modern terminology (italicized) to label concepts from that earlier time, Turing's typescript (and the published version of it in [5]) covered:

- *evaluation rules* that assign numerical values, indicative of strength or weakness, to board configurations;
- *variable look-ahead*: instead of the consequences of every possible move being followed equally far, the "more profitable moves" are "considered in greater detail than the less";
- *heuristics* guiding the search through the tree of possible moves and countermoves;
- the *minimax* principle;
- *quiescence search*: the search along a particular branch in the tree is discontinued when a "dead" position is found—a position with no captures or other developments in the offing.

Part of computer chess's appeal was the scope it offered for investigating the idea of a program learning to improve its performance. Newman described one pioneering learning technique during a discussion on intelligent machines with Turing and others (broadcast on BBC radio). In response to the question "Can machines learn to do better with practice?," Newman said:

> Yes . . . a programme could be composed that would cause the machine to do this: a 2-move chess problem is recorded into the machine in some suitable coding, and . . . a white move is chosen at random . . . All the consequences of this move are now analysed, and if it does *not* lead to forced mate in two moves, the machine prints, say, "P-Q3, wrong move," and stops. But . . . when the right move is chosen the machine not only prints, say, "B-Q5, solution," but it changes the instruction calling for a random choice to one that says "Try B-Q5." . . . Such a routine could certainly be made now, and I think this can fairly be called learning [71, p. 496].

Turing wanted to use what we would now call a genetic algorithm, or GA, to achieve learning. He hinted at the idea in a quotation given previously ("try out variations of behavior and accept or reject them"), and the idea also appeared, briefly, in a report he wrote in 1948 for the National Physical Laboratory, where he spoke of "genetical or evolutionary search," with "the criterion being survival value", [66, p. 431]. Turing fleshed out the idea in his typescript on chess:

> [A]s to the ability of a chess-machine to profit from experience, one can see that it would be quite possible to programme the machine to try out variations in its method of play (e.g. variations in piece value) and adopt the one giving the most satisfactory results. This could certainly be described as "learning," though it is not quite representative of learning as we know it [70, p. 575].

As far as can be ascertained from surviving records, Turing's chess experiments seem to have been conducted entirely with paper machines. The earliest known

[11]For details of this move see [12, pp. 395-401].
[12]*The Times*, 11 June 1949.
[13]Letter from Bowden to Turing, 23 November 1951, John Rylands Library, University of Manchester, collection GB 133 TUR/ADD/53.

implementation of a GA on an electronic computer was by Samuel, in 1955, in connection with a checkers-playing program [54]. Samuel set up two copies of his checkers-player on an IBM 704 and programmed the computer to try out variations in the method of play. It made small random alterations to the move generator of one copy, leaving the other copy unchanged; and, after a series of games, alterations that led to improved performance were retained. In this way, the program learned to outplay Samuel in "8 or 10 hours of machine-playing time" [54, p. 211].

ARTIFICIAL "NEURONS"

In his 1948 report [66] for the National Physical Laboratory, Turing discussed a raft of concepts that would later become central to AI, generally after reinvention by others. This report, titled simply "Intelligent Machinery," was in effect a manifesto.

It circulated in limited numbers, but never received the exposure it warranted. As well as Turing's Search Principle, and his all-too-brief mention of genetical search, he discussed the theorem-proving approach to AI, the concept of multiagent "cultural" searches, the role of randomness in learning and in computation more broadly, intelligent robots, the idea of intelligence as an "emotional" or response-dependent concept [51], the argument from Gödel's theorem against computer intelligence (later elaborated in [31] and [48]), and much more besides, including a cameo presentation of the Turing test (see below). Moreover, in the report's most detailed sections he set out a bottom-up brain-inspired approach to machine intelligence that foreshadowed aspects of connectionist AI [17], [60].

Turing suggested (in [66]) that practical computing systems be constructed out of simple, initially randomly connected neuron-like elements, and then trained to perform specific tasks. McCulloch and Pitts had already published an account of their model neurons (McCulloch later remarking, "What we thought we were doing (and I think we succeeded fairly well) was treating the brain as a Turing machine"), but there was no suggestion in their work of using artificial neurons to construct neural network-like computing systems [37], [38]. Moreover, their discussion of neuron-level learning was perfunctory. Turing's idea that an initially unorganized artificial neural network could be organized by what he called "interfering training" was new.

His "unorganized machines"—"A-types" and "B-types"—are species of neural networks. He said A-types are "about the simplest model of a nervous system" [66, p. 418]. A-types and B-types both consisted of interconnected Boolean neurons, and B-types had in addition inputs that enabled training.

Using these inputs, an external agent would organize the initially randomly connected network by selectively disabling and enabling connections within it, an arrangement that is functionally equivalent to one in which the stored information takes the form of new connections within the network.

Turing said that a B-type can be trained—by means of applying "appropriate interference, mimicking education"—to "do any required job, given sufficient time and provided the number of units is sufficient" [66, p. 422].

He foresaw using an electronic computer to simulate the network, and also envisaged programming an automatic training algorithm—although he himself carried out his neural-network experiments using paper machines, a short time before the first functioning electronic computers came alive. He said:

> I feel that more should be done on these lines. I would like to investigate other types of unorganised machine . . . When some electronic machines are in actual operation I hope that they will make this more feasible. It should be easy to make a model of any particular machine that one wishes to work on within such a U.P.C.M. [universal practical computing machine] instead of having to work with a paper machine as at present. If also one decided on quite definite "teaching policies" these could also be programmed into the machine. One would then allow the whole system to run for an appreciable period, and then break in as a kind of "inspector of schools" and see what progress had been made [66, p. 428].

ROBOTS AND "CHILD MACHINES"

Turing advocated a range of different approaches to developing AI—and, as the field progressed, all of his suggestions turned out to be fecund. One approach was his bottom-up neural network strategy, another was the higher level approach now called "Symbolic AI," involving his ideas about search, heuristics, theorem-proving, and high-level learning. He also contrasted the approach of programming disembodied systems—systems for carrying out some "abstract activity, like the playing of chess"—with an approach involving embodiment, saying "I think both approaches should be tried" [67, p. 463]. He thought researchers pursuing an embodiment approach could aim, for example, "to provide the machine with the best sense organs that money can buy, and then teach it to understand and speak" [67, p. 463].

Turing sketched the quest, now pursued in research labs round the globe, to build a humanoid robot:

> One way of setting about our task of building a "thinking machine" would be to take a man as a whole and try to replace all the parts of him by machinery. He would include television cameras, microphones, loudspeakers, wheels and "handling servo-mechanisms" as well as some sort of "electronic brain" [66, p. 420].

By interacting with the environment, the robot would be "finding things out for itself," he said (ibid.). The "brain," moreover, might be "stationary and control[ing] the body from a distance" (ibid.).

Turing himself steered away from an embodiment approach, saying it would be "a tremendous undertaking" and could "depend rather too much on sense organs and locomotion to be feasible" [67, pp. 420–421]. While this type of approach was certainly not suited to those early years of AI, advances in engineering made it feasible within two decades. Brooks wrote in a review of the embodiment approach's history that Turing "carefully considered the question of embodiment"; Brooks summarized the approach: "robots have bodies and experience the world directly—their actions are part of a dynamic with the world" [7, pp. 3, 4]. Brooks' Turingesque research-program at MIT produced some of AI's best-known early robots, including Herbert, Cog, and Kismet [6], [7], [8].

Another of Turing's emphases was his *child-machine* concept:

> Instead of trying to produce a programme to simulate the adult mind, why not rather try to produce one which simulates the child's? [67, p. 460].

The child-machine is to be endowed by its makers with what it needs in order to learn as a human child would. "Presumably the child-brain is something like a note-book as one buys it from the stationers," Turing said: "Rather little mechanism, and lots of blank sheets" (ibid).

> Our hope is that there is so little mechanism in the child-brain that something like it can be easily programmed. (ibid)

The child-machine might be "more or less without a body," having "at most organs of sight, speech and hearing," Turing said [66, p. 420]. Once the child machine had been "subjected to an appropriate course of education one would obtain the adult brain" [67, p. 460]. The teaching process "could follow the normal teaching of a child" [67, p. 463].

Michie was a leading advocate of Turing's child-machine concept, in AI's "classical" period (which began when the rugged pioneering era of experimental computers gave way to the mainframe world of the 1960s and 1970s). In Michie's hands, the concept led to Edinburgh University's Freddy robots [1].

The Mark two Freddy was a stationary robot with a single camera-eye and a pincer-like gripper. Michie and his colleagues taught Freddy to recognize common objects—a hammer, cup, and ball, for example—and to assemble simple objects like toy cars from a pile of parts. Although the assembly operations themselves had to be interactively programmed, the first stages of the teaching process did somewhat resemble the teaching of a child. The experimenters would spend a few hours showing the robot unfamiliar parts and demonstrating how to lay the pile of parts out ready for assembly.

Turing's far-seeing ideas on machine intelligence were being carried forward and implemented by those he influenced.

THE NEXT GENERATION

For Michie, his wartime discussions on machine intelligence with Turing were pivotal. "By the end of war I wanted to spend my life in that field," he said, "but I also knew that I would have to find something else to do while waiting until the magic moment arrived, when there were machines on which one could do suitable experimentations." Michie bided his time until the 1960s, when he set up his Experimental Programming Unit at Edinburgh.

Others, influenced by Turing in the immediate post-war years and bitten by the machine intelligence bug, did not wait so long to embark on programming. They were prepared to make do with the rough-and-ready facilities of the vanguard computers that Turing and other trail-blazing designers—such as Williams, Kilburn, Wilkes, and the team that Turing left behind at the National Physical Laboratory—had brought into existence.

Two of those experimental machines were located in Manchester and Cambridge, in Newman's Computing Machine Laboratory and Wilkes' Mathematical Laboratory. Early machine-intelligence programs came to life on those two room-sized computers, and the programmers responsible for these preliminary steps in AI were very much in Turing's orbit.

Dietrich Prinz, a refugee German physicist, took on the challenge of chess programming. He had learned how to program the Manchester computer at seminars that Turing gave, and an article by Turing's assistant Davies at the National Physical Laboratory inspired him to work on chess [16]. "To programme one of the electronic machines for the analysis of Chess would not be

difficult," Davies had written, echoing Turing in his design document for the ACE [19, p. 62].

Prinz's chess program was for solving mate-in-two problems with various simplifications imposed, such as no double-moves by pawns, and no distinction between mate and stalemate [50]. With so small a search space there was no need for Turing's heuristic approach, and Prinz wrote a brute-force program. Its first successful run was in November 1951, but Turing seemed to take little interest—perhaps because he knew there was no future in brute-force alone.

Strachey, another emerging code-hacker at Manchester—later responsible, with Scott, for denotational semantics—shared Turing's passion for mechanized learning. In 1951, he listened to one of Turing's BBC radio broadcasts on thinking machines [69] and wrote to him:

> [Y]our remark . . . that the programme for making a machine think would probably have great similarities with the process of teaching . . . seems to me absolutely fundamental . . . First . . . it would obviously be necessary to get the machine to learn in the way a child learns, with the aid of a teacher.[14]

With Turing's programming manual for the Manchester computer [68] at his elbow, Strachey coded a heuristic draughts (checkers) player. This made use of the computer's CRT monitor to display a virtual board. When the program was ready for the human player to key in his or her next move, it emitted a peremptory pip-pip sound.[15] By the summer of 1952, Strachey had developed the program to the point where it could "play a complete game of Draughts at a reasonable speed" [58, p. 47].

Also interested in language processing, he included some primitive capabilities in that direction. Like ELIZA's canned responses at MIT in the 1960s, the program's conversational output would appear at the printer. Upon the toss of a coin to decide who should move first, the program would print either "HEADS" or "TAILS" at random, and then demand "HAVE I WON?"[16] If the human hesitated too long over a move, the program might say "YOU MUST PLAY AT ONCE OR RESIGN"; and ineptitude from its opponent, or blatant rule-breaking, might elicit

> I REFUSE TO WASTE ANY MORE TIME. GO AND PLAY WITH A HUMAN BEING.

Strachey said his experience with his draughts program convinced him that a "great deal of what is usually known as thinking can in fact be reduced to a relatively simple set of rules of the type which can be incorporated into a program" [59, p. 26].

Strachey's 1951 letter to Turing[16] also described his paper-machine experiments with his NIM-player, a learning program (it remembered and tried to reachieve any winning position it reached). He did not include learning in his draughts program, however. This was done later by Samuel (see above); after Strachey publicized the program at a Canadian conference in 1952, Samuel coded a version for the IBM 701, saying "The basic program used in these experiments is quite similar to the program described by Strachey in 1952" [54, p. 212]. Samuel's program ran at IBM in late 1952, an example of pre-Dartmouth AI in the U.S.[17]

In 1951, Wilkes' computer in Cambridge presented a none-too-interested world with two programs incorporating learning. Both were written by an American visitor to the Mathematical Laboratory, Oettinger, who was fueled by Turing's ideas about mechanized learning [45, p. 1243]—and he also cited thoughts on the topic by other scientists in Britain, including Ashby and Grey Walter as well as Wilkes [2], [24], [75].[18]

Oettinger's "response-learning" program operated "at a level roughly corresponding to that of conditioned reflexes," Oettinger said [45, p. 1257]. He trained it to respond appropriately to given stimuli by means of what he described as "approval or disapproval." Turing had suggested in his 1948 manifesto that the "training of the human child depends largely on a system of rewards and punishments, and this suggests that it ought to be possible to carry through the organising with only two interfering inputs"—and had himself described some experiments along those lines involving a paper machine [66, pp. 425–429]. Oettinger summed up the results of his experiments with the response-learning program as follows (his words might call to mind Turing's remark that "we have little temptation to imagine intelligence" when "we are able to explain or predict [the] behaviour"):

> The behaviour pattern of the response-learning . . . machine is sufficiently complex to provide a difficult task for an observer required to discover the mechanism by which the behaviour of the . . . machine is determined [45, p. 1257].

[14]Letter from Strachey to Turing, 15 May 1951, King's College Archive, Cambridge.
[15]Turing's and Strachey's pioneering work on computer music is described in [15].
[16]The program's remains are in the Strachey Papers, Bodleian Library, Oxford.
[17]I benefitted from my correspondence with Samuel in 1988.
[18]I benefitted from interviewing Oettinger in 2000 and from information in a letter he wrote to me dated 19 June 2000.

Oettinger described his second exhibit, his "shopping programme," as a "child machine" [45, p. 1247]. This learned in a way reminiscent of "a small child sent on a shopping tour" [45, p. 1247]. The program's world consisted of eight shops and the user would instruct it to find a specified item. While searching, initially at random, the program memorized a few of the items stocked in each shop it visited. If sent out again for the same item, or for some other item whose location was learned during the previous searches, the program went directly to the appropriate shop.

Oettinger observed that Turing's "imitation game can be played with the shopping . . . machine" [45: p. 1250], and was it seems the first programmer to claim an implemented program capable of passing a (very) restricted Turing test—where the interrogator's questions are "restricted to shopping orders of the form 'in what shop may article *j* be found?' coded as vectors":

> Under these conditions the interrogator . . . would find it difficult to make the correct identification [45, p. 1250].

Oettinger regarded this as indicating that "machine intelligence . . . exists, although in a very limited form," and he noted (in a phrase reminiscent of Minsky's later characterization of AI) that his program was "capable of performing functions which, in living organisms, are considered to be the result of intelligent behaviour" [45, pp. 1250–51].

THE TURING TEST

Turing's test, his "imitation game," needs no introduction. Its first appearance was in Turing's 1948 manifesto, in a restricted form:

> It is not difficult to devise a paper machine which will play a not very bad game of chess. Now get three men as subjects for the experiment, A, B, and C. A and C are to be rather poor chess players. B is the operator who works the paper machine . . . Two rooms are used with some arrangement for communicating moves, and a game is played between C and either A or the paper machine. C may find it quite difficult to tell which he is playing [66, p. 431].

Turing added, "This is a rather idealised form of an experiment I have actually done" (ibid.).

His description of the unrestricted form of the test was published two years later [67, pp. 441-442]. B is now an electronic computer. C, the "interrogator," must decide based on question-and-answer (conducted via, e.g., a teleprinter), which of A and B is the computer. A should "help the interrogator." The computer is "permitted all sorts of tricks so as to appear more man-like" (as Turing explained in the script of his radio discussion with Newman [71]), and C—"who should not be expert about machines"—is permitted to ask "anything" [71, p. 495].

Turing's claims for this "question-and-answer method" are that (a) it "seems to be suitable for introducing almost any one of the fields of human endeavour that we wish to include," and (b) it "has the advantage of drawing a fairly sharp line between the physical and the intellectual capacities of a man" [67, p. 442]. He re-emphasized that point in his radio script: the "important thing is to try to draw a line between the properties of a brain, or of a man, that we want to discuss, and those that we don't" [71, p. 94].

It is worth noting that Turing has frequently been misinterpreted as intending his test as a *definition*, in particular a behaviorist or "operational" definition. This misunderstanding was introduced by early commentators; and, since then, the leitmotif that Turing attempted a definition has permeated very widely through academic and popular literature (see, e.g., [56, pp. v–vi], [4, p. 248], [29, p. 415], [21], [22], [32], [42, p. 158]). Yet Turing stated very clearly in his radio script:

> I don't want to give a definition of thinking, but if I had to I should probably be unable to say anything more about it than that it was a sort of buzzing that went on inside my head. But I don't really see that we need to agree on a definition at all [71, p. 494].

Turing also made it completely clear that his test was intended to provide a *criterion* (his term [67, p. 442]) but that this sufficient condition was not also necessary. He said:

> May not machines carry out something which ought to be described as thinking but which is very different from what a man does? . . . [A]t least we can say that if, nevertheless, a machine can be constructed to play the imitation game satisfactorily, we need not be troubled by this objection [67, p. 442].

The usefulness of Turing's 70 years old test as a serious benchmark for AI in the 21st century is certainly something that can be questioned, especially given reports of not-so-intelligent chatbots (such as Eugene Goostman[19]) passing the unrestricted Turing test.

[19]"Turing Test success marks milestone in computing history," University of Reading, 8 June 2014. https://archive.reading.ac.uk/news-events/2014/June/pr583836.html

However, it is a common feature of modern presentations of the Turing test that Turing's specification of what counts as passing the test is omitted [67, p. 441]. In the early days, though, this aspect of the test was well understood. Oettinger gave the following explanation of what is required for a computer to be said to pass:

> He [Turing] postulates a game played by a man A, a woman B, and an interrogator C . . . [T]he object of the game is for C to make the correct identification . . . If, when A is replaced by a machine, C is wrong in his identifications as often as when A was a man, the man and the machine become indistinguishable to C [45, p. 1250].

Turing himself was very clear: the question that replaces "our original, 'Can machines think?'" is:

> Will the interrogator decide wrongly as often [in the computer versus human game] as he does when the game is played between a man and a woman? [67, p. 441].

Curiously—and despite the number of Turing test competitions that have been held around the globe—this man–woman pretest, necessary for properly scoring the Turing test, seems not to have been conducted to date. Yet the pretest would not be overly challenging to carry out, and in its absence there is no protocol for properly determining whether a computer has passed Turing's test.

What did Turing say about *when* his test might be passed? He thought that "in about fifty years' time", the state of the art would have progressed sufficiently to enable a program to "play the imitation game so well that an average interrogator will not have more than 70 per cent. chance of making the right identification after five minutes of questioning" [67, p. 449]. Events proved him right about that. He placed *passing* the test much further in the future, however. Discussing this in his 1952 radio script, he responded to Newman's question whether success would "be a long time from now, if the machine is to stand any chance with no questions barred?":

> Oh yes, at least 100 years, I should say [71, p. 495].

So perhaps reports that the unrestricted Turing test has already been passed are premature?

The Eugene Goostman chatbot was subjected to a careful series of unrestricted Turing tests, held at the Royal Society of London in 2014. The outcome: Turing was correct in his statement of what would be achieved "in about fifty years"—a third of the 30 interrogators made the wrong identification after 5 minute of questioning.[18] The organizers announced though that the chatbot had therefore passed the Turing test—because they thought that according to Turing, "If a computer is mistaken for a human more than 30% of the time during a series of five minute keyboard conversations it passes the test."[18] There are others who have taken this prediction of Turing's as his intended threshold for passing the test (e.g., [53, p. 152]), but the difficulty with this interpretation is that it drives Turing into contradicting his own words. The 30% failure rate for judges would, he said, be achieved "in about fifty years," but the test would not actually be passed for "at least 100 years."

Turing's benchmark test is much harder than it might appear, and his prediction of final success in the test has at least three more decades to run. Time will tell. But as well as posing a significant challenge, the Turing test is an enduring emblem of the cocktail of ideas that Turing and company served up in those early days of the quest to build intelligent machinery.[20]

CONCLUSION

From the early 1940s, Turing contributed significantly and influentially to the theory of what we now call AI. In the postwar years, he implemented AI programs in the form of what he termed "paper machines," these including chess programs, simulations of artificial neural networks, and learning programs. His work inspired other early programmers, such as Prinz, Strachey, and Oettinger, who ran chess, checkers, and learning programs on vanguard electronic computers in 1951 and 1952. Turing's ideas on robotics and "child machines" provided a basic theoretical framework for some early robots developed in the 1960s and 1970s. His "imitation game" remains a hard challenge and even a guiding principle for aspects of AI today.[21]

BIBLIOGRAPHY

[1] A. P. Ambler, H. G. Barrow, C. M. Brown, R. M. Burstall, and R. J. Popplestone, "A versatile computer-controlled assembly system," in *Proc. 3rd Int. Joint Conf. AI*, Stanford, CA, USA, 1973, pp. 298–307.

[2] W. R. Ashby, "Design for a brain," *Electron. Eng.*, vol. 20, pp. 379–383, 1948.

[3] A. G. Bell, *The Machine Plays Chess*. Oxford, U.K.: Pergamon, 1978.

[20] Turing's test has also sparked a large literature in the philosophy journals, too vast to be surveyed here. The following references give the flavor of some current debates: [51], [23], [20], [40], and [74], with [52] responding to views in some of these papers.

[21] I am grateful to two anonymous reviewers for helpful comments.

[4] N. Block, "The computer model of the mind," in *An Invitation to Cognitive Science*, vol. 3, D. N. Osherson and H. Lasnik, Eds. Cambridge, MA, USA: MIT Press, pp. 247–289, 1990.

[5] B. V. Bowden, *Faster Than Thought*. London, U.K.: Pitman, 1953.

[6] C. Breazeal and B. Scassellati, "Infant-like social interactions between a robot and a human caretaker," *Adaptive Behav.*, vol. 8, pp. 49–74, 2000.

[7] R. A. Brooks, "Intelligence without reason," Memo No. 1293, MIT AI Laboratory, Apr. 1991. [Online]. Available: https://people.csail.mit.edu/brooks/papers/AIM-1293.pdf

[8] R. A. Brooks and L. A. Stein, "Building brains for bodies," *Auton. Robots*, vol. 1, pp. 7–25, 1994.

[9] B. E. Carpenter and R. W. Doran, "The other Turing machine," *Comput. J.*, vol. 20, pp. 269–279, 1977.

[10] D. Champernowne, letter, *Comput. Chess*, vol. 4, pp. 80–81, 1980.

[11] A. Church, review of [61], *J. Symbolic Log.*, vol. 2, pp. 42–43, 1937.

[12] B. J. Copeland, *The Essential Turing*. Oxford, U.K.: Oxford Univ. Press, 2004.

[13] B. J. Copeland, *Alan Turing's Automatic Computing Engine*. Oxford, U.K.: Oxford Univ. Press, 2005.

[14] B. J. Copeland, "The Church-Turing thesis," in *Stanford Encyclopedia of Philosophy*, E. Zalta, Ed. 2023. [Online]. Available: https://plato.stanford.edu/entries/church-turing/

[15] B. J. Copeland and J. Long, "Alan Turing: How his universal machine became a musical instrument," IEEE Spectrum, Oct. 2017. [Online]. Available: https://spectrum.ieee.org/alan-turing-how-his-universal-machine-became-a-musical-instrument

[16] B. J. Copeland and D. Prinz Jr., "Computer chess—The first moments," in *The Turing Guide*, B. J. Copeland et al. Oxford, U.K.: Oxford Univ. Press, 2017, pp. 327–346.

[17] B. J. Copeland and D. Proudfoot, "On Alan Turing's anticipation of connectionism," *Synthese*, vol. 108, pp. 361–377, 1996.

[18] M. Croarken, *Early Scientific Computing in Britain*. Oxford, U.K.: Oxford Univ. Press, 1990.

[19] D. W. Davies, "A theory of chess and noughts and crosses," *Sci. News*, vol. 16, pp. 40–64, 1950.

[20] S. Danziger, "Intelligence as a social concept: A socio-technological interpretation of the Turing Test," *Philosophy Technol.*, vol. 35, pp. 1–26, 2022.

[21] R. French, "The Turing test: The first 50 years," *Trends Cogn. Sci.*, vol. 4, pp. 115–122, 2000.

[22] A. T. Greenhill and B. R. Edmunds, "A primer of artificial intelligence in medicine," *Techn. Innovations Gastrointestinal Endoscopy*, vol. 22, pp. 85–89, 2020.

[23] B. Gonçalves, "The Turing test is a thought experiment," *Minds Machines*, vol. 33, pp. 1–31, 2023.

[24] W. Grey Walter, "Possible features of brain function and their imitation," in *Proc. Symp. Inf. Theory*, 1950, pp. 134–136.

[25] J. Haugeland, *Artificial Intelligence: The Very Idea*. Cambridge, MA, USA: MIT Press, 1985.

[26] D. Hilbert, "Über die Grundlagen der Logik und der Arithmetik," in *Verhandlungen des 3. Internationalen Mathematiker-Kongresses: in Heidelberg vom 8. bis 13. August 1904*, A. Krazer, Ed., Leipzig, Germany: Teubner, pp. 174–185, 1905.

[27] D. Hilbert and W. Ackermann, *Grundzüge der Theoretischen Logik*. Berlin, Germany: Springer, 1928.

[28] H. Hinsley, *British Intelligence in the Second World War*, vol. 2. London, U.K.: HMSO, 1981.

[29] A. Hodges, *Alan Turing: The Enigma*. London, U.K.: Vintage, 1992.

[30] S. Lavington, Ed., *Alan Turing and His Contemporaries*. Swindon, U.K.: Brit. Inform. Soc., 2012.

[31] J. R. Lucas, "Minds, machines and Gödel," *Philosophy*, vol. 36, pp. 112–127, 1961.

[32] J. Lotman, "The phenomenon of culture," in *Juri Lotman—Culture, Memory and History*, M. Tamm, Ed., Cham, Switzerland: Palgrave Macmillan, pp. 33–48, 2019.

[33] P. Mahon, "History of Hut 8," Bletchley Park, 1945, in [12], pp. 267–312.

[34] P. Mancosu and R. Zach, "Heinrich Behmann's 1921 lecture on the decision problem and the algebra of logic," *Bull. Symbolic Log.*, vol. 21, pp. 164–187, 2015.

[35] J. McCarthy, "Information," *Sci. Amer.*, vol. 215, pp. 64–73, 1966.

[36] P. McCorduck, *Machines Who Think*. New York, NY, USA: Freeman, 1979.

[37] W. McCulloch, discussion, in *John Von Neumann: Collected Works*, vol. 5, A. H. Taub, Ed., London, U.K.: Pergamon Press, pp. 319–328, 1961.

[38] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *Bull. Math. Biophys.*, vol. 5, pp. 115–133, 1943.

[39] D. Michie, *On Machine Intelligence*, 2nd ed. Chichester, U.K.: Ellis Horwood, 1986.

[40] P. Millican, "Alan Turing and human-like intelligence," in *Human-Like Machine Intelligence*, S. Muggleton and N. Chater, Eds. Oxford, U.K.: Oxford Univ. Press, pp. 28–51, 2021.

[41] M. Minsky, Ed., *Semantic Information Processing*. Cambridge, MA, USA: MIT Press, 1968.

[42] L. Moody and W. K. Bickel, "Substance use and addictions," in *Computer-Assisted and Web-Based Innovations in Psychology, Special Education, and Health*, J. K. Luiselli and A. J. Fischer, Eds. Amsterdam, Holland: Elsevier, pp. 157–183, 2016.

[43] A. Newell, J. C. Shaw, and H. A. Simon, "Empirical explorations with the logic theory machine: A case study in heuristics," in *Proc. Western Joint Comput. Conf.*, 1957, vol. 15, pp. 218–230.
[44] A. Newell and H. A. Simon, "Computer science as empirical inquiry: Symbols and search," *Commun. Assoc. Comput. Mach.*, vol. 19, pp. 113–126, 1976.
[45] A. G. Oettinger, "Programming a digital computer to learn," *Philos. Mag.*, vol. 43, pp. 1243–1263, 1952.
[46] C. S. Peirce, "The 1903 Lowell Institute lectures I–V," in *Charles S. Peirce* (part 2), vol. 2, A.-V. Pietarinen, Ed., Berlin, Germany: de Gruyter, 2021.
[47] C. S. Peirce, "Some amazing mazes [conclusion]," *Monist*, vol. 18, pp. 416–464, 1908.
[48] R. Penrose, *Shadows of the Mind: A Search For the Missing Science of Consciousness*. Oxford, U.K.: Oxford Univ. Press, 1994.
[49] G. Polya, *How To Solve It*. Princeton, NJ, USA: Princeton Univ. Press, 1945.
[50] D. G. Prinz, "Robot chess," *Research*, vol. 5, pp. 261–266, 1952.
[51] D. Proudfoot, "Rethinking Turing's test," *J. Philosophy*, vol. 110, pp. 391–411, 2013.
[52] D. Proudfoot, "An analysis of Turing's criterion for 'thinking'," *Philosophies*, vol. 7, pp. 1–15, 2022.
[53] W. Rapaport, "Turing test," in *Encyclopedia of Language and Linguistics*, 2nd ed., K. Brown, Ed., Boston, MA, USA: Elsevier, pp. 151–159, 2006.
[54] A. L. Samuel, "Some studies in machine learning using the game of checkers," *IBM J.*, vol. 3, pp. 211–229, 1959.
[55] C. E. Shannon, "Programming a computer for playing chess," *Philos. Mag.*, vol. 41, pp. 256–275, 1950.
[56] C. E. Shannon and J. McCarthy, Eds., *Automata Studies*. Princeton, NJ, USA: Princeton Univ. Press, 1956.
[57] S. W. Skan, *Handbook For Computers*, (2 vols). London, U.K.: Dept. Sci. Ind. Res., 1954.
[58] C. S. Strachey, "Logical or non-mathematical programmes," in *Proc. Assoc. Comput. Machinery*, 1952, pp. 46–49.
[59] C. S. Strachey, "The thinking machine," *Encounter*, vol. 3, pp. 25–31, 1954.
[60] C. Teuscher, *Turing's Connectionism*. London, U.K.: Springer, 2001.
[61] A. M. Turing, "On computable numbers, with an application to the Entscheidungsproblem," *Proc. London Math. Soc.*, series 2, 1936-37, vol. 42, pp. 230–265. Reprinted in [12], pp. 58-90 (page references in the text to this edition).
[62] A. M. Turing, "Systems of logic based on ordinals," *Proc. London Math. Soc.*, series 2, 1939, vol. 45, pp. 161–228. Reprinted in [12], pp. 146-204 (page references in the text to this edition).
[63] A. M. Turing, "The steckered Enigma. Bombe and Spider," Bletchley Park, 1940, in [12], pp. 314–335.
[64] A. M. Turing, "Proposed electronic calculator," National Physical Laboratory, 1945, in [13], pp. 369–454.
[65] A. M. Turing, "Lecture on the Automatic Computing Engine" delivered to the London Mathematical Society, 1947, in [12], pp. 378–394.
[66] A. M. Turing, "Intelligent machinery," National Physical Laboratory, 1948, in [12], pp. 410–432.
[67] A. M. Turing, "Computing machinery and intelligence," *Mind*, vol. 59, pp. 433–460, 1950. Reprinted in [12], pp. 441-464 (page references in the text to this edition).
[68] A. M. Turing, *Programmers' Handbook for Manchester Electronic Computer Mark II*. Manchester, U.K.: Computing Machine Laboratory, Univ. Manchester, circa 1950. [Online]. Available: https://archive.computerhistory.org/resources/text/Knuth_Don_X4100/PDF_index/k-4-pdf/k-4-u2780-Manchester-Mark-I-manual.pdf
[69] A. M. Turing, "Can digital computers think?," BBC Third Programme, 1951, in [12], pp. 482–486.
[70] A. M. Turing, "Chess," typescript, 1951, in [12], pp. 569–575.
[71] A. M. Turing, R. Braithwaite, G. Jefferson, and M. Newman, "Can automatic calculating machines be said to think?," BBC Third Programme, 1952, in [12], pp. 494–506.
[72] M. Y. Vardi, "Who begat computing?," *Commun. ACM*, vol. 56, p. 5, 2013.
[73] T. Vickers, "Applications of the Pilot ACE and the DEUCE," in [13], pp. 265–279.
[74] M. Wheeler, "Deceptive appearances: The Turing test, response-dependence, and intelligence as an emotional concept," *Minds Machines*, vol. 30, pp. 513–532, 2020.
[75] M. V. Wilkes, "Can machines think?," *Spectator*, no. 6424, pp. 177–178, 1951.
[76] J. R. Womersley, "A.C.E. project – Origin and early history," National Physical Laboratory, 1946, in [13], pp. 38–39.

B. JACK COPELAND is distinguished professor in humanities at the University of Canterbury, Christchurch, 8041, New Zealand. He received his doctorate in mathematical logic from Oxford University, and has written and edited 7 books concerning Turing, most recently The Turing Guide (Oxford University Press, 2017). Contact him at jack.copeland@canterbury.ac.nz.