A clarification on Turing's test and its implications for machine intelligence

Phil Maguire (pmaguire@cs.nuim.ie)
Philippe Moser (pmoser@cs.nuim.ie)
Department of Computer Science
National University of Ireland, Maynooth

Rebecca Maguire (rebecca.maguire@ncirl.ie)
School of Business, National College of Ireland
IFSC, Dublin 1, Ireland

Abstract

Turing's (1950) article on the Turing test is often interpreted as supporting the behaviouristic view that human intelligence can be represented in terms of input/output functions, and thus emulated by a machine. We show that the properties of functions are not decidable in practice by the behaviour they exhibit, a result, we argue, of which Turing was likely aware. Given that the concept of a function is strictly a Platonic ideal, the question of whether or not the mind is a program is a pointless one, because it has no demonstrable implications. Instead, the interesting question is what intelligence means in practice. We suggest that Turing was introducing the novel idea that intelligence can be reliably evidenced in practice by finite interactions. In other words, although intelligence is not decidable in practice, it is *testable* in practice. We explore the connections between Turing's idea of testability and subsequent developments in computational complexity theory.

Keywords: Turing test, functionalism, mathematical objection, artificial intelligence, Chinese room argument, P versus NP.

Introduction

The Turing test (Turing, 1950) is often interpreted as suggesting that the mind can be viewed as a program, and that intelligence can, in effect, be emulated by a machine. For example, Hodges (2013) states that a "fair characterisation" of the implicit assumptions in Turing's paper is the idea that all physical action is, in effect, computable. He also states that Turing's underlying argument was that "the organisation of the brain must be realisable as a finite discrete-state machine." Penrose (1994) presents Turing's argument as that of viewing "physical action in general - which would include the action of a human brain - to be always reducible to some kind of Turing-machine action." Such views imply that Turing would have supported the philosophy of functionalism, the idea that the mind can be viewed as a function delivering a particular mapping between input and output.

We will show that, in fact, the properties of functions cannot be decided in practice by the behaviour they exhibit. Hence, the observation that all physical actions are computable is a useless one. Physical action is finite, meaning that any such action will always have the potential of having been produced by trivial mechanical means. This article could, in principle, have been written by chance by an algorithm selecting random characters. It's possible. Alternatively, you could chat for an hour with somebody over video link and only later find out that what you thought was a live link was nothing more than a historical recording being played back. The scenario is unlikely, yet it remains a possible explanation for any observation of finite behaviour. According to Aaronson (2006): "In principle, we can always simulate a person by building a huge lookup table, which encodes the person's responses to every question that could ever be asked...So there's always *some* computational simulation of a human being". The observation that "physical action is always reducible to some kind of Turing-machine action" (Hodges, 2013) is therefore a misleadingly simplistic characterisation of Turing's (1950) work.

We suggest that Turing's idea was not about machines emulating intelligence, but about the possibility of intelligence being evidenced *in practice*. His point was that, although intelligence is a Platonic ideal (i.e. cannot be decided in practice), it is somehow manifested in finite objects, meaning that finite tests can detect it with high confidence. Hodges (2009) succinctly expresses this alternative idea: "operations which are in fact the workings of predictable Turing machines could nevertheless appear to the human observer as having the characteristics of genuine intelligence and creativity".

The first argument we will make is that, as regards the mind-program debate, Turing supported the mathematical objection and held the intuitive view that the mind could not be a program.

The second argument we will make is that Turing (1950) was pointing out that the mind-program debate concerns Platonic ideals and hence is not relevant in practice. At the same time he was also conjecturing that although intelligence is not decidable in practice, it is somehow *testable* in practice.

Turing and the mathematical objection

Lucas (1961) and Penrose (1994) have argued that Gödel's first incompleteness theorem shows that the human mind cannot be a program. Lucas makes the observation that, given any formal system which claims to emulate the mind, its Gödel sentence can be produced mechanically by a Turing machine. People can 'see' that this sentence is true, but the system cannot, meaning that there is at least one thing humans can do that formal systems cannot. Penrose (1994) develops this argument using Turing's theorem, arguing that no program can fully account for the set of all humanly accessible methods for ascertaining mathematical truth. These ideas fall into the category of argument which Turing (1950) characterises as "the mathematical objection":

"There are a number of results of mathematical logic which can be used to show that there are limitations to the powers of discrete-state machines...The short answer to this argument is that although it is established that there are limitations to the powers of any particular machine, it has only been stated, without any sort of proof, that no such limitations apply to the human intellect."

One requirement for Lucas's argument to succeed is that human minds are consistent, the explicit assertion of which would seem to be ruled out by Gödel's second incompleteness theorem. Lucas (1976) responds by suggesting that "we must assume our own consistency, if thought is to be possible at all..." From this perspective, consistency is not something to be established, but rather the starting point for understanding.

Another assumption in Penrose's (1994) use of Turing's theorem is that Church and Turing's conception of effective method (i.e. computation) is genuinely universal. This assumption, known as the Church-Turing thesis, does not appear to be something that can be proved in the traditional sense. Nevertheless, Church considered his description of computation as a definition, and even Turing was convinced of its veracity. According to Turing (1954): "The statement is...one which one does not attempt to prove. Propaganda is more appropriate to it than proof, for its status is something between a theorem and a definition."

As one of the founders of the discipline of computer science, Turing was in a good position to evaluate the mind-program debate. Although he notes in 1950 that assertions of the mind's superiority are "without any sort of proof", adherence to the mathematical objection is a consistent theme of his writings. In his 1938 PhD thesis, carried out under the supervision of Church, Turing makes clear that the mind has an intuitive power for performing uncomputable steps beyond the scope of a Turing machine:

"Mathematical reasoning may be regarded rather schematically as the exercise of a combination of two faculties, which we may call intuition and ingenuity. The activity of the intuition consists in making spontaneous judgments which are not the result of conscious trains of reasoning...In consequence of the impossibility of finding a formal logic which wholly eliminates the necessity of using intuition, we naturally turn to non-constructive systems of logic with which not all the steps in a proof are mechanical, some being intuitive."

After the Second World War, Turing's view on the role of intuition in reasoning appears unchanged. In a 1948 report to the National Physical Laboratory, Turing again clarifies that mathematicians' ability to decide the truth of certain theorems appears to transcend the methods available to any Turing machine:

"Recently the theorem of Gödel and related results...have shown that if one tries to use machines for such purposes as determining the truth or falsity of mathematical theorems and one is not willing to tolerate an occasional wrong result, then any given machine will in some cases be unable to give an answer at all. On the other hand 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".

In his last article published before his death in 1954, Turing again emphasises the role of intuition beyond effective method. He argues that Gödel's theorem shows that 'common sense' is needed in interpreting axioms, something a Turing machine can never demonstrate:

"The results which have been described in this article are mainly of a negative character, setting certain bounds to what we can hope to achieve purely by reasoning. These and some other results of mathematical logic may be regarded as going some way towards a demonstration, within mathematics itself, of the inadequacy of 'reason' unsupported by common sense."

Separating the Platonic and the practical

Turing's writings reveal him to be a consistent proponent of the mathematical objection. Nevertheless, it was not something he ever attempted to prove. Instead, in 1950 he made a surprising statement: "I do not think too much importance should be attached to it".

As we will show, the question of whether the mind is a program is unrelated to the question of whether physical machines can demonstrate intelligent behaviour. A problem with the associated philosophical debate is that it confuses Platonic ideals, such as 'function' and 'program', with concepts that can be manifested in practice through finite means, such as mechanical machines. For example, Searle (1980) seeks to address a question concerning Platonic ideals (is the mind a program?) via a thought experiment involving a practical mechanism (his Chinese room argument).

To be clear, when Turing (1936) refers to a 'machine', he is referring to a Turing machine, or program, not a physical machine. His idea of a rule-following automaton represents what a human following instructions would be able to achieve using pen and paper. Turing machines cannot be built in practice. For a start, they require an infinitely long read/write tape. They also require infinite mechanical precision, with the head being able to move up and down the tape in perpetuity without its position slipping. All of the 'computers' that we encounter in everyday life are merely physical finite state machines which are projections of a Platonic ideal. Accordingly, when we use the word 'machine' in this article, we are referring to a finite physical machine which can be realised in practice. The Platonic concept of a rule-following automaton we refer to as a 'Turing machine' or 'program'.

In his 1950 article, Turing is referring, not to Turing machines, but to finite physical machines. He is wondering about the practical implications of the Platonic theory of computation. For example, can the mathematical objection be demonstrated to have any observable practical implications?

Below, we provide a modification of the use theorem (see Oddifreddi, 1992) to show that no finite set of interactions is sufficient for checking the behaviour of a black-box system: properties of functions cannot be decided in practice. No matter how many questions we are allowed to ask, it is not possible to deduce that the black-box system is capable of solving a given problem. As stated informally by Block (1981), "two systems could be exactly alike in their actual and potential behaviour, and in their behavioural dispositions and capacities and counterfactual behavioural properties...yet there could be a difference in the information processing that mediates their stimuli and responses that determines that one is not at all intelligent while the other is fully intelligent".

More formally, let O be an observer, A a set of strings, and $f : \Sigma^* \to \{0,1\}$ be a Boolean function. O can adaptively ask finitely many queries $f(x) = ?$ (O has access to A), after which O decides whether f computes the set A, i.e. $f(x) = A(x)$ for every x. The following standard argument shows every observer is wrong on some function.

Formal argument *For any observer O, and any set A, there is a function* $f : \Sigma^* \to \{0,1\}$ *such that O is wrong on f.*

Proof. Let O be as above, A be a set. If O rejects all functions (i.e. thinks all functions do not compute A) then O is wrong on f, where $f(x) = A(x)$ for every x. So let g be accepted by O. O queries g on finitely many strings $x_1, x_2, \ldots, x_n$. On all the strings $x_1, x_2, \ldots, x_n$, g is equal to A, otherwise O is wrong about g. Choose y different from $x_1, x_2, \ldots, x_n$, and construct $f : \Sigma^* \to \{0,1\}$, by letting $g(x) = f(x)$ for all $x \neq y$, and $f(y) = 1 - A(y)$. f does not compute A, because f is different from A on input y. Because f equals g on inputs $x_1, x_2, \ldots, x_n$ (the ones queried by O), O will make the same decision about f than about g, i.e. O decides that f can compute A. By construction of f, O is wrong.

In summary, not only is a Turing-style test not capable of deciding intelligence, a finite sequence of input/output is not even sufficient for deciding the program that computed it. Accordingly, we can see that, while Searle's (1980) Chinese room argument presents a valid observation of the limitations of Turing-style tests for deciding the origins of strings, it cannot possibly say anything about the mind-program debate. The separation, or lack of, between minds and programs has no practical implications that could be exposed by a Chinese room scenario.

Because our argument uses a similar diagonal proof to Turing's (1936) theorem, it seems likely that he was aware of the general idea. If the properties of functions cannot be decided in practice, it is clear that the relevant real-world question is not about establishing once and for all whether the mind is a program, but about what intelligence means *in practice*. This is the issue that, we believe, Turing (1950) was addressing.

Intelligence in practice

Searle (1980) makes the case that the automaton in the Chinese room, which intuitively appears to have no understanding, would "pass the Turing test", thus supposedly refuting the idea that the test can decide if a system is intelligent. We suggest that this is a straw man argument. Turing never made any statements about the 'passing' of his test, or about its capacity to reliably decide the emulation of human understanding.

Turing seeks merely to establish the possibility of "satisfactory performance" at the imitation game over a finite period; not perfect performance, nor the idea that satisfactory performance is a reliable indicator of subsequent perfect performance. He never goes beyond claims for the finite simulation of intelligence: "My contention is that machines can be constructed which will simulate the behaviour of the human mind very closely" (Turing, 1951).

One source of misunderstanding is that Searle and Turing use different meanings for the word 'thinking'. For Searle, a machine can be described as thinking when it is capable of emulating human performance. For Turing, a machine can be described as thinking when it successfully simulates a finite set of human performance. While Searle treats 'thinking' as a Platonic ideal, Turing identifies that, in practice, there can be nothing to thinking beyond acting indistinguishably from the way a thinker acts.

Dismissing the mathematical objection

Whereas Turing's 1936 article established the Platonic concept of computation, his 1950 paper investigates the practical implications of this concept. There are two components to his conjecture. The first is that the mathematical objection (i.e. a possible separation between programs and minds) has no implications in the finite, physical world. The second is that intelligence does have certain practical implications, which are somehow *testable*.

Dealing with the first component, Turing (1948) notes that the mathematical objection based on Gödel's theorem rests on a proviso that the Turing machine is not allowed to make mistakes. However, "the condition that the machine must not make mistakes...is not a requirement for intelligence". In other words, Turing is pointing out that, although physical machines cannot emulate intelligence, they can, when engineered with sufficient precision, simulate it to any desired level. At the limit, mistakes are inevitable, but *in practice* those mistakes can be pushed back as far as one wants. Turing (1947), in his earliest surviving remarks concerning AI, points out that this would allow machines to play very good chess:

"This...raises the question 'Can a machine play chess?' It could fairly easily be made to play a rather bad game. It would be bad because chess requires intelligence. We stated... that the machine should be treated as entirely without intelligence. 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."

Rather than dismissing the mathematical objection, Turing (1950) conjectures that it does not result in practical limitations: in the physical world there will always be some machine which is up to the job of simulating intelligence to a required level. The mathematical objection is a Platonic rather than practical one:

"There would be no question of triumphing simultaneously over all machines. In short, then, there might be men cleverer than any given machine, but then again there might be other machines cleverer again, and so on."

At first blush, this withdrawal of the mathematical objection appears to eliminate the possibility of evidencing intelligence in practice. For instance, if stupid machines can simulate any finite set of behaviour, and pass any test, then it can be argued that behaviour alone is never sufficient for establishing intelligence. Nothing we can do in the real world, no behaviour we can perform, can offer conclusive evidence of non-trivial origin. Could it be that intelligence is useless?

This is exactly the attitude adopted by Professor Jefferson in his 1949 Lister Oration, whom Turing (1950) cites: "Not until a machine can write a sonnet or compose a concerto because of thoughts and emotions felt, and not by the chance fall of symbols, could we agree that machine equals brain - that is, not only write it but know that it had written it."

Here, Jefferson is arguing that finite behaviour alone is not sufficient for establishing intelligence. Instead, we must 'know' what strings mean and 'feel' emotions. Because such properties can never be represented symbolically, there is no possibility of any system, human or otherwise, evidencing its intelligence in practice. But this doesn't seem right. Intuitively, our intelligence is something useful. It lets us achieve things in the real world that less intelligent systems cannot. The big question is whether there is there any reliable test that can somehow validate this intuition regarding the practical utility of intelligence.

What Turing (1950) is pointing out is that, although intelligence can never be emulated in practice, it must somehow be possible to evidence it in practice with high confidence. For example, it seems feasible that a finite signal beamed from a distant solar system could convince us that it harbours intelligent life. Granted, we could never be absolutely 100% sure, but it seems plausible that there exist signals that could lead us to be very, very confident.

Indeed, all the communication we have ever had with other human beings can be summarized as a finite string of symbols. If intelligence could not be evidenced in practice through finite interactions, it would preclude humans from identifying each other as intelligent, reducing us to solipsists. According to Aaronson (2006), "people regularly *do* decide that other people have minds after interacting with them for just a few minutes...there *must* be a relatively small integer *n* such that by exchanging at most *n* bits, you can be reasonably sure that someone has a mind".

It seems that in order for the concept of intelligence to be meaningful, there must be some practical means of identifying and engaging with intelligent systems in the real world. Having realised this, Turing (1950) remarks "I am sure that Professor Jefferson does not wish to adopt the extreme and solipsist point of view. Probably he would be quite willing to accept the imitation game as a test."

Testing for intelligence

Some have interpreted Turing (1950) as suggesting that infinite testing is required to establish intelligence, spread over an infinite space of time (e.g. Harnad, 1992). But, again, this conceptualisation of intelligence as being infinitely distant holds no value, because it never delivers practical results. Instead, Turing is saying something far more significant. He is saying that, although intelligence is not something that can be decided, it is something that can be reliably *tested*.

Let us consider the question of "what is a test"? A test is of finite duration. Applying it to an object yields results that enable inferences to be drawn about that object. Somehow, the results hold significance for other aspects of the object, beyond those which have been directly tested. One could say that the test succeeds in succinctly 'characterising' the object through a finite set of responses.

For example, students are asked to sit tests to reveal how much they know about a particular subject. Because of the short duration, it is not possible to ask them every question that could possibly be asked. Instead, questions are chosen cleverly so that responses can be relied on to draw inferences about students' ability to answer all the other potential questions which haven't been asked.

Of course, a particular student might get lucky on a test. They might fortuitously have learned off the answers to the exact questions which came up, but no others. Thus, as we have already shown, a test can never *decide* whether a student understands a subject. What a cleverly crafted test *can* do is offer a very high level of confidence that the student would have answered other questions correctly.

What are the properties of a good test that would lead us to have such confidence? In short, a good test is one for which there is no easy strategy for passing it, other than full mastery of the subject. For a start, there should be no way for the student to get a copy of the test in advance, or predict what will be on it so that they can just learn off the relevant responses. In addition, the test should be well diversified, bringing together material from many different areas of the subject. For instance, the answers should draw on different aspects of understanding and not betray a simple pattern which allow them to all be derived using the same technique. Furthermore, to be as hard to compute as possible, successive answers should be integrated with each other, rather than addressing totally separate chunks of knowledge.

The next question is whether testing for a property can continue to yield dividends in terms of raising confidence closer to certainty. For example, it seems intuitive that a short two-hour test can provide a clear picture of a student's ability in a subject. But what if we extended the length of the test to three hours? Can a test be designed such that it continues to build confidence continually higher? Is it conceivable that there are some properties, such as intelligence, that would support continuous testing of this nature to any level of confidence? Such a mechanism could 'bridge' the gap between the Platonic and physical worlds and render the concept of intelligence meaningful in practice. We propose that this is the fundamental premise underlying Turing's (1950) conjecture.

Opening the door for machine intelligence

An important implication of Turing's testability is that it paves the way for machines to display intelligent behaviour. Tests are finite. The ability to pass hard tests can therefore be encoded in finite machines, which are themselves hard to construct and hard to understand, yet still feasible in practice.

In a BBC radio debate in 1952, Turing connects the idea of 'thinking' with this capacity to do things which are reliably difficult. Even when the mechanics of a machine are exposed, it can still retain the ability to do 'interesting things', which are not rendered trivial by the overt mechanisation. In other words, explicitly finite objects can still pass hard tests:

"As soon as one can see the cause and effect working themselves out in the brain, one regards it as not being thinking, but a sort of unimaginative donkey-work. From this point of view one might be tempted to define thinking as consisting of 'those mental processes that we don't understand'. If this is right then to make a thinking machine is to make one which does interesting things without our really understanding quite how it is done."

When Jefferson confuses this concept of hardness with the difficulty of identifying the implementation, Turing immediately corrects him:

"No, that isn't at all what I mean. We know the wiring of our machine, but it already happens there in a limited sort of way. Sometimes a computing machine does do something rather weird that we hadn't expected. In principle one could have predicted it, but in practice it's usually too much trouble. Obviously if one were to predict everything a computer was going to do one might just as well do without it."

What Turing (1952) is getting across is that finite objects can have a form whose mechanical implications are in principle predictable, but in practice are hard to anticipate. We know exactly what the program is, we can see its structure, yet its relationship with potential input is a complex one. Even when the validity of a machine's responses can be easily verified, and we can see it computing the answers, the reason the machine works can still be hard to fathom, other than doing the calculations oneself. This property is what, for Turing, constitutes 'thinking'.

What this implies is that hard tests may actually be hard to find. When we put forward what appears to be a challenging test for AI, such as chess, we cannot know for sure how hard it is. As soon as a machine succeeds in defeating the test, the associated limitations become apparent. At that point we go back to the drawing board to develop a harder test. Yet the process never ends. In the same way that it is not possible to decide intelligence, it is not possible to decide the reliability of a test for intelligence. Tests must themselves be tested.

This explains why Turing (1950) was upbeat on the imminent prospect of artificial intelligence. No matter what elaborate tests we conceive of, there will always be feasible machines that succeed in passing them: "...there might be men cleverer than any given machine, but then again there might be other machines cleverer again, and so on." The surprising success of IBM's Deep Blue over chess champion Gary Kasparov in 1997 can be seen as a vindication of Turing's principle of unending testability. According to Turing, no matter how we shift the goalposts for intelligence tests in the future, we will never be able to rule out the possibility of machine success. The intuition that intelligence must somewhere trump machine will remain simply that: an intuition.

Computational complexity theory

In 1956 Gödel wrote a letter to the dying von Neumann, echoing Turing's remarks on a potential gap between the Platonic theory of computation and its practical implications. In the letter Gödel identified a finite analogue of the Entscheidungsproblem which Turing (1936) originally addressed by demonstrating the existence of uncomputable problems. Gödel realised that, although this uncomputability must kick in at the Platonic limit, it did not necessarily apply in practice for deciding the existence of solutions of finite length. This would present the possibility of using an algorithm to quickly decide if a given mathematical statement had a proof of feasible length. He explained to von Neumann that this "would have consequences of the greatest importance" because "the mental work of a mathematician...could be completely replaced by a machine".

These novel ideas that Turing and Gödel struggled to express have since developed into a field known as computational complexity theory. This discipline now provides a framework that can be used to formally define concepts such as 'smart questions' and 'high confidence', which are integral to the Turing test. Smart questions are, for example, those that involve solving instances of an NP-hard problem (e.g. computing a Hamiltonian path or a subset-sum solution).

In 1950 Turing didn't have the formal tools needed to express these ideas, he was relying on his intuition. What is interesting is that many of the key questions in computational complexity theory, such as that raised by Gödel in 1956, continue to lie unresolved. For example, it not yet known if there are problems whose solutions can be easily verified, yet are hard to compute. Do smart questions really exist? Can hard tests be created that engender high confidence from finite responses? This is known as the P versus NP problem, which remains the biggest unsolved problem in computer science today. While computational complexity theory has succeeded in formalising the key components in Turing's conjecture, which concern the interaction between the Platonic and practical domains, it has not yet succeeded in answering the difficult questions that ensue. Aaronson (2006) eloquently sums up the impasse: "All ye who would claim the intractability of finite problems: that way lieth the P versus NP beast, from whose 2^n jaws no mortal hath yet escaped".

Conclusion

Interpretations of Turing's (1950) work have focused strongly on the idea of running the test. The article has often been interpreted either as being supportive of functionalism (e.g. Searle, 1980), or of advocating a trite, deeply flawed test for evaluating the intelligence of artificial systems through the process of imitation (e.g. French, 2012).

In this article we have argued that Turing (1950) was neither claiming that the mind is a program, nor providing a heuristic for evaluating the progress of AI resting on human psychology. Instead, Turing was making the observation that although the mathematical objection collapses in practice, it is somehow possible for intelligence to be evidenced with high confidence through finite interactions.

Philosophers such as Searle (1980) have confused Turing's definition of 'thinking' with the emulation of intelligence. We have shown that the question of whether the mind is a program is not one that has implications in the real world. Any finite set of behaviour could have been produced by a trivial process that simply outputs the behaviour from a database or produces it randomly. Turing's key idea is that, although intelligence is not decidable in practice, an observer's confidence in testing for intelligence can increase quickly with the length of the interaction. In other words, our intelligence provides us with a means of quickly posing and responding to finite tests which are reliably hard to pass. At the core of this conjecture lies the idea that intelligence gives us the ability to quickly and easily verify, with high confidence, properties that appear hard to compute.

Because it was not possible for Turing in 1950 to present his conjecture mathematically, he instead chose to publish these ideas as a philosophical article in the journal Mind. An unfortunate outcome of this choice is that Turing's (1950) article seems whimsical. Reading it quickly, one might almost imagine that Turing was playing the gender-based imitation game at a party and stumbled by chance upon the idea of using it as a test for intelligence. Hayes and Ford (1995) go so far as to suggest Turing was proposing "a test of making a mechanical transvestite" and state that "Turing's vision from 1950 is now actively harmful to our field".

Turing's trite presentation betrays the sophisticated theory behind the concept. From his extended writings we can see that he was concerned with, not the idea of a psychological standard for AI, but the more general concept of how intelligence can be evidenced in practice. In particular, Turing (1950) was not claiming that every test that humans come up with is reliable. Inevitably, if a Turing-style test is run using laypeople, the programs that get furthest will be those that exploit the weaknesses of human psychology. Turing's conjecture instead concerns the actual concept of testability - the idea that if a group of world-leading experts got together and laboured for long enough, they would be able to distil stronger and stronger tests for intelligence, though without ever being able to *decide* the reliability of a test. The assumption that Turing's (1950) concept can be addressed by a localised testing event involving untrained and unsophisticated judges is thus a serious misinterpretation of the basic idea.

In conclusion, we have proposed that the Turing test does not aim to decide a yes/no answer to the question of whether or not a system is intelligent. It does not address, and was never intended to address, the question of whether the mind is a program. The Turing test is the observation that finite interactions can result in very high confidence in a system's ability to exhibit intelligent behaviour. Hence, intelligent 'thinking' machinery is feasible in practice.

References

Aaronson, S. (2006). PHYS771 lecture 10.5: Penrose.

Block, N. (1981). Psychologism and behaviorism. *The Philosophical Review*, 5–43.

French, R. M. (2012). Moving beyond the Turing test. *Communications of the ACM*, *55*(12), 74–77.

Harnad, S. (1992). The Turing test is not a trick: Turing indistinguishability is a scientific criterion. *ACM SIGART Bulletin*, *3*(4), 9–10.

Hayes, P., & Ford, K. (1995). Turing test considered harmful. In *Ijcai (1)* (pp. 972–977).

Hodges, A. (2009). *Alan Turing and the Turing test*. Springer.

Hodges, A. (2013). Alan Turing. In E. N. Zalta (Ed.), *The stanford encyclopedia of philosophy* (Winter 2013 ed.).

Lucas, J. R. (1961). Minds, machines and Gödel. *Philosophy*, *36*(137), 112–127.

Lucas, J. R. (1976). This Gödel is killing me: A rejoinder. *Philosophia*, *6*(1), 145–148.

Odifreddi, P. (1992). *Classical recursion theory: The theory of functions and sets of natural numbers*. Elsevier.

Penrose, R. (1994). *Shadows of the mind* (Vol. 52). Oxford University Press Oxford.

Searle, J. R. (1980). Minds, brains, and programs. *Behavioral and brain sciences*, *3*(03), 417–424.

Turing, A. M. (1936). On computable numbers, with an application to the Entscheidungsproblem. *J. of Math*, *58*(345-363), 5.

Turing, A. M. (1947). Lecture on the ACE.

Turing, A. M. (1948). Intelligent machinery.

Turing, A. M. (1950). Computing machinery and intelligence. *Mind*, 433–460.

Turing, A. M. (1951). Intelligent machinery, a heretical theory.

Turing, A. M. (1954). Solvable and unsolvable problems.

Turing, A. M., Braithwaite, R., Jefferson, G., & Newman, M. (1952). Can automatic calculating machines be said to think?

Turing, A. M., & Copeland, B. J. (2004). *The essential Turing: seminal writings in computing, logic, philosophy, artificial intelligence, and artificial life, plus the secrets of Enigma*. Clarendon Press Oxford.