Edmundo Balsemão Pires
Burkhard Nonnenmacher
Stefan Büttner-von Stülpnagel
Editors

Relations of the Self

IMPRENSA DA UNIVERSIDADE DE COIMBRA
COIMBRA UNIVERSITY PRESS

COIMBRA 2010

Edo Pivčević
Bristol

TURING MACHINE AS AN IDEA OF AN INTELLIGENCE WITHOUT CONSCIOUSNESS

I

A Turing machine represents a model of a serial step-by-step computational procedure for solving problems that can be expressed in digital terms. What is known as the Church-Turing thesis amounts to the claim that *any* computable function is computable by such a machine. Indeed there is a widely held view, shared, incidentally, by Turing himself, that computability by a Turing machine defines what 'computable' means. This view, though widely held, is nevertheless not uncontested, and in recent times in particular there has been a great deal of talk about the so-called 'super-recursive algorithms' which are claimed by some mathematicians to provide more efficient models for solving certain problems. However I do not propose to engage in this particular debate. What is relevant to my present purpose is that such criticism as is directed against Turing's computational model is prompted mainly by the apparent inability of such a model to provide answers to certain questions arising specifically in connection with what is known as the 'halting problem' – the questions which, I shall argue, require a reference to a conscious agent.

But more on this later. As far as Alan Turing himself was concerned, he did not seem to have any doubts that intelligent reasoning in all its forms, essentially, represented certain algorithmic operations; in particular such as could be executed mechanically by what he called a 'discrete state' machine. In other words, he took what was in effect a reductionist view of consciousness. With intelligent activity being computational in nature, he seemed to think, consciousness, strictly speaking, was operationally redundant. This was surprising in a way, for, as we shall see presently, he himself had proved an important mathematical theorem, which appeared to undercut his own reductionist argument by showing in effect that there was a world of difference between the questions that can be *meaningfully asked* and the questions that can be *computationally answered*; a difference which presupposes a conscious activity of understanding.

2

Consider, first, his views on the computational nature of thought as set out in his paper *Computing Machinery and Intelligence* (1950), which has inspired a vast body of research into Artificial Intelligence.

To ask whether machines can 'think', and hence mimic human intelligence, as Turing points out, inevitably raises the problem of definition of 'machine' and 'think', and here the ordinary usage is not necessarily a reliable guide. The only way to make the question sufficiently precise, he argues, is to rephrase it in operational terms; more specifically, in terms of what he calls an 'imitation game'. In his own version of the game (many alternative versions have been proposed since) two persons, a man and a woman, are interrogated via a teleprinter by a third player, who sits in a separate room and tries to determine from the answers given – some of which may be deliberately misleading – which of the other two belongs to what sex. Turing's main thesis is that such rational substance as may be extracted from the question 'Can machines think?' can in this instance be expressed operationally by asking: 'If one of the two players being quizzed is replaced by a sufficiently powerful digital computer, would the interrogator come to a wrong decision as often as before?' An alternative, and somewhat simplified, version of the game involves an interrogator quizzing a computer and a human, both separated from the interrogator by a screen, in an attempt to establish which one is which. If, after prolonged questioning, the interrogator is unable to decide which of the two examinees is a machine and which is human then, Turing argued, there can be no valid reason for denying that machines are capable of the same kind of intelligence as we ourselves are.

The objective of the 'imitation game' was thus to demonstrate that the capacity for rational thought can to all intents and purposes be treated as a capacity for a specific computational manipulation of information. Once it is established – if it can be established – that operationally there is no essential difference between the capacity for reasoning and a computational manipulation of information any remaining hesitations one might have in accepting that machines can think can only be due to prejudice. What is more, one will have to concede that machines are in principle capable of performing any activities in which rational thinking processes play a part, such as setting up and evaluating hypotheses, drawing inferences from past experience, correcting their own mistakes, and generally adapting their behaviour to suit specific tasks.

3

Yet it may be questioned if the 'imitation game' really does show that the intellectual activity of the human brain can be defined purely 'computationally', i.e. in a mechanical fashion, without presupposing, or referring to, consciousness. Surely, it might be argued, the capacity we have of reflectively monitoring what we say, feel or do, not only influences the way we act but is a precondition of *understanding* of what is actually going on at the time. Even if the machine involved in the 'imitation game' produced correct answers on every

single occasion, there would still remain an important difference between its operations and the thinking processes of the human examinee, for such processes involve insight and understanding, and insight and understanding presuppose consciousness. But, then, how can thinking processes be literally compared to algorithmic operations performed by an automaton?

Turing's reply to this is that if the machine's *actual performance* matches that of a human being, the question of consciousness need not bother us unduly. As it happens, there is, he points out, something closely similar to the imitation game which has been around for a long time, namely *viva voce* examination. In such an examination the examinee's grasp of the acquired information is tested in the light of the answers he/she gives irrespective of what may or may not be going on 'inside their heads'. It is the actual performance that matters; all else is immaterial.

Still the uneasiness persists, and it is not altogether surprising that it should. The reason for such uneasiness was already articulated by Plato, who addressed this issue in his discussion of knowledge. What Plato was anxious to point out was that having knowledge, or being in possession of knowledge, in the *true* sense of the word, implies an understanding of what one claims to know. But if that was the case, he argued, knowledge could not be equated simply with correct opinion or belief, even though the difference between them may not be that important in practical terms. Thus a man who knows the way to Larissa and another man who merely has the correct opinion as to which route to take will both guide travellers successfully to their destination. Correct opinion is thus 'no less useful for action' than knowledge. Yet, insists his Socrates - they are *not* the same. What is more, although there were few things he (Socrates) would be inclined to claim he really knew, this was most definitely one of them.[1]

Plato's point, in short, is that while one can guess, or, by various means be induced to accept, correct opinions or beliefs, one cannot be made to see that – let alone *understand the reasons why* – such opinions or beliefs are correct. True knowledge cannot be had without understanding, and understanding cannot be mechanically imparted; and hence, strictly speaking, not (not conclusively, at any rate) externally verified either, even though the conditions can be created, in particular by skilful instruction, where an insight hopefully might emerge. It is one thing to guess at the truth, Plato was in effect saying, and quite another to know it. Theoretically all of one's opinions could be correct opinions, but they would not count as genuine knowledge unless they were accompanied by understanding and insight, and neither of these are possible without consciousness.

If, however, consciousness is involved in 'knowing correct answers', then surely the functionalist theory of mind of the kind that Turing was advocating fails to give the full story. Now at this point, no doubt, the objection will again be raised that what matters in the end is not what goes on 'inside one's head' on some given occasion but whether, and how, it is possible to establish and maintain a successful exchange of information. What is important, it will be said, is whether the questions asked, or the answers given, are meaningful and contextually appropriate, even if the answers in the given case may not be correct.

---

[1] *Meno*, 98b

But that, of course, is precisely the problem. For how do we decide if the questions asked and the answers given are indeed meaningful and appropriate? A decision here clearly will depend a great deal on what goes on 'inside our head'. Ordinary language abounds with concepts that are intelligible only with a reference to what goes on 'inside our heads'. So either we shall have to devise an entirely different kind of language for transacting information, a language, that is, which has been systematically cleansed of any such references – in which case we shall have to confront the problem of justifying the loss of meaning involved in translations of ordinary language statements into such a reformed language – or else we shall have to abandon the reductionist policy and reconcile ourselves to what we instinctively know to be true, viz. that there is more to the use of language than can be mimicked by any computational manipulation of symbols. The point briefly is this: that if by some miracle a successful and sustained communication in terms of ordinary language concepts could be established with a computational mechanism then the conclusion would have to be not that the concept of consciousness is redundant but that the mechanism in question is *endowed with consciousness*.

4

Turing's advocacy of what was in effect a reductionist computational theory of intelligent mental processes is even more remarkable, considering that an inspiration for much of his work came from a mathematician whose proof of the inescapable incompleteness of any (consistent) system which encompasses ordinary arithmetic is sometimes seen as having dealt a severe blow to the ambitions of any such theory, i.e. Kurt Gödel. It is true that Gödel's proof from time to time has been subject of extravagant claims, implying, among other things, that algorithmic procedures were not necessarily involved in arriving at mathematical truth, and that some other mysterious, or at any rate as yet unidentified processes were here at work. Nevertheless, as we shall see, his result provided a useful reminder that there was a place for mathematical intuition, flowing in particular from an understanding – and hence involving *conscious awareness* – of what one is doing when engaging in the activity of computation. Turing, by contrast, came to pursue a completely opposite route, dismissing any ostensibly non-computational aspects of computational activity as irrelevant and incidental to intelligent thought.

I shall come to Gödel's proof in moment. First, let us consider briefly a theorem which Turing proved in his paper 'On computable numbers, with an application to the *Entscheidungsproblem*' (1936)[2] and which, as I have already said, appears to run counter to the computational model of mind he subsequently came to espouse.

It was, incidentally, in this paper that he first introduced the concept of a universal computing machine (subsequently known as the universal Turing machine) – a kind of universal operating system capable of running any program, or computing any mathematical algorithm. He did this, interestingly enough, with the expressed purpose of showing that

[2] Reprinted in Stephen Hawking's anthology of seminal mathematical papers *God Created the Integers* (2006).

there were limits to what could be achieved by mechanical computation. He wanted to prove, that is, that there could be no universal computational procedure whereby it might be established unequivocally with respect to any given algorithm (any number-generating computational device) if that algorithm, for any given input, will or will not successfully complete the operation and generate what is recognisably a 'computable number'. 'Computable numbers', according to his definition, were 'the real numbers whose expressions as a decimal are calculable by finite means'; more clearly, they were numbers whose decimal digits could be successively worked out by a finite number of elementary mathematical operations on a finite number of symbols. Thus e.g. the square root of two evidently satisfies this condition, even though its decimal expansion is non-periodic as well as infinite. By contrast, incomputable numbers by definition cannot be expressed (calculated) in this way. A central implication of Turing's proof was that there could be no universal mechanical test for deciding in any given case if the relevant mathematical proposition ascribing a property to a certain number, or numbers, is or is not a theorem – i.e. whether it is provably true or false.

In simple words, the problem Turing addressed might be expressed thus: Is it possible to establish *mechanically* what can and what cannot be established *mechanically*? That this turns out to be impossible is of some considerable philosophical significance.

5

Let us look briefly at how he arrived at this result. Having defined what he called 'computable numbers', his next step was to consider the possibility of a mechanical procedure whereby all algorithms generative of such numbers might be listed in such a way that each of them could be assigned a unique identity tag, or a unique 'description number'. The question, in short, was whether it was possible to *computably order* such algorithms? His reasoning owes a great deal to the so-called 'diagonal procedure' devised by G. Cantor, with a view to showing that there exist sets which are 'non-denumerable', i.e. which are greater than, and hence cannot be put into one-to-one correspondence with, the ('countable') set of natural numbers. What Cantor proved with the help of his 'diagonal procedure' was that the set of all real numbers (i.e. rationals + irrationals) in particular falls into this category, and therefore belongs to a higher order of infinity.

Suppose we try to list all real numbers between 0 and 1 by correlating them with the set of integers, thus:

1. 0. $a_1,a_2,a_3\ldots$
2. 0. $b_1,b_2,b_3\ldots$
3. 0. $c_1,c_2,c_3\ldots$

and so on

Then by picking out the sequence $a_1,b_2,c_3\ldots$ which runs along the diagonal of the above list, and by suitably altering the respective digits as we go along, we can ensure that the resulting new sequence – i.e. the 'diagonal number' – will differ from any other number included in the list in *at least one place.* It follows that the set of all real numbers is not 'listable'; i.e. there are more real numbers than there are integers.

Now in a similar way Turing proposes to show that the set of all 'computable numbers' (in the sense of his definition) is not recursively, or computationally, listable. His proof involves the following key steps. Consider again a universal ('Turing') machine capable of operating any algorithm. Assuming that all algorithms (all computational mechanisms, that is) could be listed numerically, then clearly each of them could be attached a unique 'description number', such that if a given 'description number' were to be inserted into such a machine the machine would dutifully start churning out the relevant sequence.

The question that now presents itself is this: Could there be an algorithm capable of producing a list of 'description numbers' of *all computable reals*? The answer to this, it turns out, is no, for (this was the essence of his proof) if such an algorithm existed it could be combined with a modified universal Turing machine to generate a 'diagonal number' which (as in Cantor's argument) has a 'description number' which is *not* in the list. In other words, there can be no mechanical procedure for sorting out 'description numbers' of algorithms which do from those which don't express computable numbers; end of story.

Why is all this important? In order to understand what is here at stake it might be helpful to recall some well-known views of Leibniz. Leibniz was firmly of the opinion that every problem, not just in mathematics but in all other disciplines too (except possibly in theology) could in principle be solved algorithmically, viz. by way of calculation, provided it was articulated in an appropriately reformed precise language (although, unlike Turing, he apparently did not draw from this any reductionist inferences with regard to consciousness). *Calculemus*, let us calculate, was his advice and his guiding principle. 'Thus I assert' – he wrote – 'that all truths can be demonstrated about things expressible in this [reformed] language with the addition of new concepts not yet expressed in it – all such truths, I say, can be demonstrated *solo calculo*, or solely by manipulation of characters according to certain form, without any labour of the imagination or effort of the mind, just as occurs in arithmetic and algebra.'[3]

Turing's proof, together with some other related results obtained by K. Gödel, A. Church and E. Post, in effect spells the end of this Leibnizian dream. What emerges is that there can be no universal computational decision procedure, no single universal inferential calculus – or what Leibniz called *calculus ratiocinator* – which might enable us to decide in any given case whether the relevant proposition is or is not a theorem.

And this of course leaves us with a problem of truth: for although there can be no universal computational decision procedure for determining which arithmetical propositions are and which are not provable, it is still meaningful to ask if a proposition whose truth value cannot be thus decided (i.e. decided by the available computational methods) might nevertheless be true? The problem, as I pointed out earlier, is that the range of what can be meaningfully asked, and consequently the range of what can be meaningfully asserted, does not literally reduce to what in any given instance can be computationally decided. But, then, the inevitable question is, how can a 'reductionist' computational model of mind be adequate?

[3] Cf. E. Bodemann: *Die Leibniz-Handscriften der Königlichen öffentlichen Bibliothek zu Hannover*; quoted in Benson Mates, *The Philosophy of Leibniz* (New York - Oxford, 1986, p. 185n.) from where the above translation is taken.

6

Time has now come to take a closer look at Gödel's 'incompleteness proof', which towers in the background of all such reflections, and which is often said to have contributed more than any other piece of mathematical or philosophical reasoning towards exposing the essential defectiveness of the reductionist model of mind which Turing eventually embraced.

For Gödel the problem that initially presented itself was whether mathematics could be fully axiomatised, such that all its theorems, i.e. all mathematical truths, could be demonstrated, or 'derived', from a given (finite) set of axioms and definitions. Or to put it in another way, whether there is a consistent system with a clearly specifiable axiomatic base consisting of a finite number of premises, from which it can be unambiguously decided in any given case if the relevant mathematical expression is or is not a theorem. What Gödel conclusively showed was that this was a logical impossibility. It is sufficient for a formalised system to include the ordinary arithmetic of integers for it to remain irredeemably incomplete; in other words, it will always be possible by using the arithmetical resources available within such a system to construct perfectly valid theorems which, however, will not be formally demonstrable from its axioms. One might try to increase the power of such a system by adding new axioms to its axiomatic base, but, provided the system remains consistent, no matter how far its axiomatic base is expanded it will never be able to encompass the *entire range* of mathematical truth. In other words, there will always be true mathematical propositions which will remain just outside of what can be formally demonstrated in any given instance. On the assumption that such a system is consistent, it will necessarily be incomplete.

The assumption of consistency is vital, for in an inconsistent system anything can be proved. In short, what Gödel's incompleteness proof shows is that consistency is essentially incompatible with completeness, and *vice versa*. No formalised consistent system of general arithmetic can be complete, i.e. there will always be true mathematical propositions which will be formally undecidable from its axioms. At the same time, it is not possible to demonstrate the consistency of such a system (i.e. show that no contradictions flow from its premises) by relying on its resources alone. This did not mean that no proofs of consistency were possible, only that in this case no 'internal' proof of consistency could be provided.

Gödel's proof (as do many other mathematical proofs) exploits the idea of self-reference. Gödel has namely devised a method of representing meta-mathematical statements about arithmetical expressions in terms of arithmetical expressions themselves. This method of 'arithmetisation' of meta-mathematics involved assigning a unique number to every symbol or sequence of symbols in a formalised system of arithmetic, including statements to the effect that a given mathematical sentence can, or for that matter cannot, be derived from the given axioms, i.e. that it is, or is not, 'provable' (and hence *decidable*) within the system. What Gödel managed to show was that a perfectly valid sentence could be constructed within such a system which says *of itself* that it is not provable, i.e. that it does *not* follow from the axioms, but which by meta-mathematical reasoning can nevertheless be shown to be *true*. But if there are true sentences which cannot be derived, or demonstrated, within such a formalised system then such a system is *incomplete*. Moreover such a system is *essentially* incomplete, because, as we saw earlier, no matter how much its axiomatic base is

expanded it will always be possible to construct new true sentences which cannot be proved within it.

In short, we are once again forced to conclude that the range of what can be meaningfully asserted, and hence the scope of possible truth, does not reduce to what within a specified mechanical system (irrespective of how powerful such a system might be) can be algorithmically decided, and this has obvious implications for a theory of mind. Nevertheless certain important qualifications need to be added to this, as will become evident in a moment.

7

Inevitably this raises important questions about the nature of mathematical truth. What procedures should be followed in establishing mathematical truth, i.e. in deciding whether a given mathematical proposition is or is not a theorem? According to Alfred Tarski 'a decision method must be like a recipe, which tells one what to do at each step so that *no intelligence* (my italics – E.P.) is required to follow it; and the method can be applied by anyone so long as he is able to read and follow directions'.[4]

Turing's view is no different. And yet, as Gödel's proof seems to show, it is not possible to settle the truth-value of *all* mathematical propositions computationally from a fixed number of axioms. In other words, there are limits to what can be computationally delivered on any given (finite) set of premises. The question of truth, it seems, raises issues which whilst not excluding computational activity nevertheless *transcend* the capacity of any specific algorithm, and can be settled only by a machine equipped with the kind of self-monitoring activity normally associated with consciousness. So either we give up the notion that mathematics is concerned with truth or we shall have to accept that consciousness is an essential ingredient in mathematical reasoning.

That mathematics is very much concerned with truth is the view of Roger Penrose, for whom the significance of Gödel's proof consists precisely in exposing a discrepancy between mathematical truth and computability (in the sense in which both Gödel and Turing employ the latter concept, i.e. step by step calculating procedure that can be mimicked by digital machines). 'If thinking' – writes Penrose – 'is just carrying out a computation of some kind, then it might seem that we ought to be able to see this most clearly in our mathematical thinking. Yet, remarkably, the very reverse turns out to be the case. It is within mathematics that we find the clearest evidence that there must actually be something in our conscious processes that eludes computation.'[5]

This of course is the exact opposite of what Turing was suggesting in his 1950 paper (i.e. *Computing Machinery and Intelligence*), to which I referred earlier. If the intellectual activity of the human mind could be explained in computational terms, Penrose is in effect saying, then the universal Turing machine might indeed be an appropriate model to use; but if

[4] 'A Decision Model for Elementary Algebra and Geometry', quoted in Raymond L. Wilder: The Foundations of Mathematics, 1958, p. 261

[5] *Shadows of the Mind*, p. 64

Gödel's proof is correct, then it must be accepted that the human mind is capable of insights which exceed the capacity of any such machine.

The reason for this, Penrose claims, is quite simple. Consider the way we go about proving, by using e.g. a Turing machine, whether a given mathematical proposition is a theorem. We instruct the machine to execute a specific algorithm, i.e. a computational procedure, or program, which hopefully will give us a definitive answer. In fact, we may never be able to obtain an answer, for the process of computation conceivably may never stop. The proposition in question, namely, may be such that its proof, or disproof, for that matter, requires an infinite number of steps, or infinite number of computational operations. So the question is, how do we establish in the given instance whether the machine will terminate its operations at some stage or whether it will go on computing forever?

Suppose we wish to decide computationally if the proposition that no number is a sum of its factors is true, i.e. whether it is a theorem, and instruct the machine to perform an appropriate computation until it finds a counter-instance. After a few steps the machine will stop at number 6, which disproves the proposition, and the problem is resolved. But suppose we wish to test the proposition that any number can be represented as a sum of four square numbers, and instruct the machine to halt when it finds a counter-instance. In this case the machine will never come to a stop: we know this because there is a perfectly valid proof (by Lagrange) which shows that the proposition is indeed true. The machine, that is, will go on forever stolidly searching for counter-instances where we know from other sources that there could be none. The point briefly is that the machine does not know what it *cannot* do, i.e. it cannot identify the questions that cannot be computationally answered.

Now superficially it might seem that this could be remedied by attaching to the machine a special computational device, a sort of meta-algorithm, which would be able to perform just such a task, i.e. decide if and when the process of computation will terminate. But what kind of algorithm exactly might be able to perform this task and provide the proof we need? There is one well-tried and highly successful method of proving mathematical theorems, namely via mathematical induction. Briefly, mathematical induction is a procedure whereby it can be shown that a given property F which attaches to a specific number attaches to all numbers. It works like this: one first assumes that F attaches to the first number of the series, and then proceeds to show that *if* F attaches to any given number n, it also necessarily attaches to its immediate successor n+1, and therefore to *all* numbers in the series. Here, of course, it should be born in mind that unlike the inductive method employed in natural sciences, which allows only of probable inferences, mathematical induction, by contrast, is conclusive, and produces proofs which cannot be overturned.

Is this, then, the kind of decision mechanism that we are looking for? Might it be possible, that is, to use mathematical induction in order to resolve the 'halting problem', i.e. decide if in the given case the process of computation by a Turing machine will or will not come to a stop? Not so, claims Penrose, and this too follows from Gödel's proof. The point is that no computational rule or algorithmic procedure is involved in judging the truth of a sentence which says of itself that it is not provable within the system, whereas mathematical induction essentially reduces to a mechanical algorithm. In short, the kind of meta-mathematical insight that enables us to decide whether in the given case the machine will, or

will not terminate its operations, 'lies beyond anything that can be formalised as a set of rules'.[6]

Penrose's conclusion, accordingly, is that mind in certain fundamental respects is not computational, and consequently cannot be mimicked by a Turing machine. If ascertaining truth or falsity of a mathematical proposition were simply a matter of performing a particular computation then the Turing machine would be able to do the job splendidly. But a decision about truth or falsity, he argues, cannot be simply a matter of computation. It demands an understanding, and hence the awareness of what the computation is about, and the machine has no understanding of what it is doing. It has no wherewithal, as it were, to enable it to grasp the meaning of its own operations, and this 'says something very significant about the mental quality of understanding'.[7]

8

Yet one has to enter here some reservations. While it is certainly true that understanding requires consciousness it does not follow from this that understanding, therefore, is non-computational. To say that a decision about truth of falsity of mathematical propositions cannot be *simply* a matter of computation is one thing, but to claim that such a decision does not necessarily require, or presuppose computation is something quite different. The first does not entail the second. What Gödel's proof shows is *not* that there are mathematical theorems which can be established non-computationally, only that the grasp of the general import of such theorems presupposes consciousness. Gödel's proof, in a sense, can itself be executed by a Turing machine – what such a machine cannot prove is the generality of its conclusion, viz. that in *any and every* system containing ordinary arithmetic, provided such a system is consistent, it will be possible to construct formally indecidable but nevertheless true mathematical sentences. In other words, not all mathematical truths can be established from a single (finite) set of axioms. Nevertheless this does not testify to a 'non-computational' nature of mathematical reasoning; it only shows that to the extent to which a Turing-type mechanism lacks a self-referential device such as is normally associated with consciousness it lacks the capacity to draw *general inferences* from it its own operations.

This is the key point, and I will return to it presently. The fact is that Penrose's 'anti-computational' argument is itself based on an algorithm. Consider the method he relies on to underpin his conclusions, i.e. *reductio ad absurdum*. This method involves an attempt to demonstrate a proposition by showing that its negation produces a contradiction. The assumption, that is, is that in order to prove a proposition it is sufficient to show that its negation cannot be consistently entertained. This is a familiar logical device widely employed in classical mathematics.

Now although the *reductio* method of proof is a powerful tool in the mathematical armoury it is not viewed with favour by some mathematicians, mainly because the tendency

[6] Ib. p. 72

[7] Ib. p. 76

often is to use it to justify certain existential inferences about numbers, which cannot be corroborated in terms of what can actually be constructed, or is even constructible in principle. This is where mathematical constructivists part company with the platonistically inclined mathematicians (like Penrose) who take the view that mathematics is essentially about 'discovering' mathematical entities and theorems that are in a sense already *there* rather than bringing them into existence through the activity of 'construction', and who therefore have no hesitation in availing themselves of the *reductio* method of proof.

But without going into the details of the dispute between constructivism and Platonism, the question that may legitimately be asked is this: assuming that the computational procedures executed by a Turing machine are not sufficient to produce the kind of proofs we need, does this necessarily mean that we have to abandon the principle of 'algorithmic' reasoning altogether? Why cannot the *reductio ad absurdum* itself be treated as a kind of algorithm; a typically *conceptual* algorithm (and hence involving conscious processes), but an algorithm nonetheless? The point is that in all arguments, there are certain definitions to be observed, certain assumptions to be made and tested, certain rules of inference to be followed. Whilst none of this, admittedly, may literally be explicable in terms of purely mechanical manipulations of symbols, it certainly does *involve* operations with symbols, and in so far as such operations are conducted in accordance with certain rules, there is no reason why they cannot be conducted by a machine.

In the end, it seems, it will all depend on what we understand by 'computation' and 'machine'. There is an automatic assumption underlying all reductionist arguments, which is that computations such as might be performed by a mechanism similar to a Turing machine are merely mechanical operations that do not need to involve consciousness. The machine is said to operate a given algorithm, or algorithms, 'blindly', i.e. without an awareness of what it is doing. But while all problem solving activity involve certain algorithmic operations it does not follow that all such operations, including the kind of logical reasoning involved in mathematical proofs, must therefore be reducible to a purely mechanical computation. That all algorithmic operations are reducible to mechanical computations reflects a specific view of algorithms. Algorithms, that is, are defined *literally* as the kind of rules that can be blindly executed by a Turing machine. But, as I have been trying to show, algorithmic reasoning, in as much as it involves manipulations of concepts, has a broader meaning, and there is every reason for supposing that such reasoning must involve consciousness. As a matter of fact, we know that such an 'algorithmic' machine does exist, namely our brain. So the main issue is not whether machines can think, but whether such things as insight and consciousness are in all cases *operationally redundant*, and it is enough to pose this question, it seems, in order to know the answer.

9

To sum up: the 'computational' approach is essentially reductionist. The key initial premise is that mental concepts are not descriptive of some mysterious mental states but have to do with certain kinds of operations involving transmission and processing of information.

In short, rather than talking in terms of 'consciousness' or 'mind' we should talk about intelligent behaviour; and such behaviour, which in various degrees is shared by all living beings, can in principle be successfully simulated and operationally tested by an algorithmic mechanism, as illustrated by Turing's 'imitation game'. Consciousness has no clearly identifiable function to perform in this process. If a machine can pass a Turing test it makes no difference if we can call it 'conscious'.

By contrast the anti-computational position involves the claim that as long as mathematical reasoning aims at discovering mathematical truth it cannot be wholly computational. Such reasoning depends on understanding and insight, and understanding and insight are not explicable in computational terms. Since they typically involve conscious processes, it follows that consciousness cannot be interpreted simply as a by-product of computational activity either. Some philosophers take this to mean that consciousness is not only non-computational but cannot be explained in physical terms at all, while others (for example Penrose) take the view that although conscious phenomena cannot be simulated on a computer, no matter how powerful, they are nevertheless manifestations of certain physical (quantum) events in the brain, even though the resources of present day physics are inadequate to explain them.

Both the computational and the non-computational position can be supported by powerful arguments, and that of course is the problem; for if both are right, neither is. The point is that the issue cannot be resolved in terms in which it is phrased. What we should be asking is not whether mathematical understanding can or cannot be explained in terms of computation, but under what conditions that which mathematics does makes sense? What in particular makes concepts such as the concept of proof, or the concept of a theorem, intelligible?

In short, while it is in principle possible to perform any algorithmic operation 'unthinkingly', it is not possible to know 'unthinkingly' why such operations are valid or invalid, why certain moves are sound while others are logically disallowed. Such questions cannot be a matter for mathematics alone. There are many concepts that form an integral part of normal mathematical reasoning but which transcend the horizon of mathematics in the narrow sense of the word. The concept of truth is particularly troublesome – so much so, in fact, that some mathematicians have argued it is best left out of mathematics altogether; with 'true' in all cases being replaced by 'derivable from certain (stipulated) axioms and definitions'. But even if this were possible – and platonistically inclined mathematicians hotly dispute that it is – this would still leave the problem of explaining the grounds of validity of mathematical propositions as well as the problem of clarifying a host of other ideas, including the distinction between that which is and that which isn't computable, the distinction between what is computable and what must logically follow, etc. – all of which require for their clarification a recourse to extra-mathematical as well as mathematical resources. If, on the other hand, the concept of truth cannot be entirely dispensed with, then it soon becomes clear that, in addition, we shall need a whole battery of other concepts with which the concept of truth is analytically linked, and that a decision in a given case as to the correct or incorrect application of such concepts cannot be a matter of mechanical manipulation of symbols alone.