Brit. J. Phil. Sci. **52** (2001), 671–682

Building Infinite Machines

E. B. Davies

ABSTRACT

We describe in some detail how to build an infinite computing machine within a continuous Newtonian universe. The relevance of our construction to the Church-Turing thesis and the Platonist-Intuitionist debate about the nature of mathematics is also discussed.

1 Introduction

Scientists and philosophers have long been fond of considering thought experiments and hypothetical situations as a means of throwing sharper light on problems in their fields. This paper follows in that tradition, and emphasizes the differences between a purely mechanical continuous Newtonian universe and our own. In particular it appears that one can build infinitely powerful computers in such a universe, and solve problems in arithmetic by the brute-force testing of an infinite number of cases in a finite period of time. The possibility of such a construction has been hinted at before, but nobody has tried to write out the details of how to do it (Boolos and Jeffrey [1989]; Hogarth [1996]).

In Section 4 we discuss why our machine is not in conflict with standard impossibility arguments from information theory. It might be thought that the possibility of constructing our machine has serious implications for the Intuitionist approach to mathematics. We take a different line in Section 5, namely that any philosophical conclusion which may be reached from our analysis is dependent upon hidden philosophical assumptions about the nature of mathematics.

A simple infinity machine is defined by Earman and Norton ([1996]) to be a computer which can carry out an infinite number of computations within a

finite length of time, by performing the individual computations faster and faster, in the manner of the Zeno paradox. In their paper they discuss the literature on this topic and demonstrate the possibility of producing such a machine in certain exotic relativistic spacetimes, sometimes called Hogarth-Malament spacetimes. In this paper we describe how to construct such a machine in a continuous Newtonian universe. By this we mean a universe obeying Newton's laws in which matter may be subdivided more and more finely while retaining the same properties. Our machine depends upon advanced nanotechnology but does not involve bodies moving with arbitrarily high speeds, the presence of an infinite amount of energy within a finite volume, or infinitely strong materials. In order to avoid considerations relating to electricity or magnetism we assume that the machine is mechanical. Its operation depends upon the movement of interlocking components, much like the computer designed, but never built, by Charles Babbage. Indeed we believe that our machine is consistent with any physics known in the year 1850.

Of course in the real universe matter is atomic, and it will be clear that our machine could not be constructed within it. The fact that our universe is also quantum mechanical at small enough scales provides an independent barrier to the construction of the machine. However, several of the objections to the construction of our machines apply with equal force to Turing machines. We argue that the rejection of *our* machines on the grounds of physical implausibility should imply the same attitude towards Turing machines. We conclude that one version of the Church-Turing thesis makes assumptions about the physics of the universe and is not provable by logical arguments alone. In some respects the real world is *less* peculiar than a purely mechanical continuous Newtonian universe would be.

2 The construction

It is known that machines involving infinite numbers of parts with no lower limit on their sizes can give rise to paradoxes (Earman and Norton [1998]), so we specify how our machines work in some detail. We suppose that it is possible to construct a machine M_1 with the following properties. M_1 contains a finite, Babbage-type computer with a specified clock time c_1 and a memory of size m_1 bytes. It also contains a robotic factory which can produce a new version of the computer and of the factory, in other words a new machine M_2. The new machine M_2 is not identical to the old one. It is supposed to have a memory of size $m_2 = 2m_1$ and its components are supposed to be 16 times smaller than those of M_1. We are thus led to the assumption that it has size (diameter) at most $s_2 = s_1/8$. If we also assume that its clock time is $c_2 = c_1/8$ then signals do not need to travel any faster in

M_2 than they did in M_1. Qualitatively speaking M_2 is smaller, faster and more powerful than M_1.

The key problem facing the construction is whether a machine can indeed produce a scaled-down version of itself. Of course we are talking about a continuous Newtonian universe so evidence from our own is suspect, but that is what is available. A poor argument in favour is that we regularly produce smaller copies of ourselves, namely babies. However, geneticists would point out that our key component, the reproductive DNA, is not scaled-down at all. The size and structure of DNA is determined by quantum theory, so the analogy is flawed. We have no idea what life might be like in a continuous Newtonian universe nor whether it would be possible.

A better argument refers to the steady improvement of machine tools over the last two centuries. Each generation of tools has been used to produce the next one, and as time has passed the tools have got steadily smaller, faster and more accurate. Most machine tools are moreover essentially classical in their operation. Similar remarks apply to computer design and performance over the period; this is a relevant observation since each generation of computers and machine tools is used to design and build the next. We see no obvious inconsistency in making the above assumption, and so feel able to continue.

The machine M_2 is capable of building M_3 and so on. So we have a potentially infinite hierarchy of machines with parameters obeying $c_{n+1} = c_n/8$, $m_{n+1} = 2m_n$, $s_{n+1} = s_n/8$. The potential total size of all the machines together is at most

$$s_1 \sum_{n=0}^{\infty} 8^{-n} = 8s_1/7 < \infty.$$

Various other quantities below are also sums of geometric series, but we will not write down the series explicitly.

We can also keep track of the time taken to manufacture the machines. Suppose t_n is the time needed to manufacture M_n. The components of M_{n+1} are 16 times smaller than those of M_n so if the tools within M_n move at the same speed as those within M_{n-1} they should build the components 16 times faster. On the other hand the memory of M_{n+1} is twice the size of that of M_n so the overall time taken may only be 8 times shorter. Thus we assume that $t_{n+1} = t_n/8$.

The assumption of these scaling factors is not as harmless as it might seem. In order to move a component of mass M a distance S in a time T from rest the force needed is of order $F \sim MST^{-2}$. Assuming that $M \sim S^3$, i.e. constant density, we deduce that the force is of order S^4T^{-2} while the pressure on a face is of order S^2T^{-2}. Thus if S and T are reduced in proportion, the

pressures on the faces of the components do not increase and we do not need the material used to be infinitely strong.

Another problematic matter is how to design the computers in such a way that the memory can be doubled indefinitely. The memory might need to have a tree-like structure in order to avoid its local structure becoming more and more complicated, on the distance scale of each machine. Or it might be designed in the classical manner of a Turing machine with a single linear tape, but of finite length.

We next discuss communication between the machines. Each of the machines is supposed to be built next to the previous one and we assume that M_n only communicates with M_{n-1} and M_{n+1}. There are two reasons for this. If n is large one cannot expect M_n to be able to report directly to M_1 since it can only transmit bits which are very short and have very small energies, and one should not expect M_1 to be able to detect these. In the reverse direction M_1 could only report to M_n extremely slowly, by the standards of M_n, because of its much longer clock time, and such reports can be carried out equally well by using the connecting chain of machines. Since M_{n+1} has twice the memory of M_n there will be data lists which M_{n+1} cannot transfer to M_n. On the other hand M_n can transfer all of its data to M_{n+1}. Because of the reducing size of the machines, the time taken to transfer data down the chain, assuming the speed of transfer is constant, also decreases exponentially with n, even though the amount of data transferred may increase at each stage.

3 The operation

We are concerned with the existence of at least one solution to an infinite sequence of problems, each of which can be resolved in a finite length of time. An example would be Fermat's last theorem. One lists the 4-tuples (a, b, c, x) of positive integers in some order and for each of them asks whether $x > 2$ and $a^x + b^x = c^x$. So in this case a positive solution to the problem corresponds to the statement that Fermat's last theorem is false (with apologies to Andrew Wiles). This gives a sequence of propositions $\mathcal{P}_n$ whose solutions take steadily more computation as n increases, because the integers involved become steadily longer when expressed in decimal or binary notation. In this paragraph we assume that the length of the computations does not increase rapidly with n. The machine M_1 is given the general problem, and told the following procedure. It is to try to solve the problem for $n = 1$. If it succeeds it is to report YES. Otherwise it constructs the machine M_2 and passes to it the entire problem together with the number $n = 2$. The second machine behaves similarly. At the nth stage the machine M_n tries to solve the problem for the case n and if it succeeds it informs the machine M_{n-1} which passes the message up the chain. If it fails it constructs

the next machine and passes the problem down the chain. After a suitable finite length of time, the first machine either hears a report that the problem is soluble for some n or can deduce from the convergence of the geometric series concerned that it is not soluble for any n.

As n increases the problems associated with storing the number increase, but we specified that the memory of each machine should be double that of the previous one, so the value of n can indeed be passed on. In addition the computation becomes longer with n at a rate over which we do not have control. In the last paragraph we assumed that this did not cause problems, but in general, by modifying the procedure as follows, we can allow the length of the computations to increase with n arbitrarily rapidly.

The program is written so that if $\mathcal{P}_n$ is being tested by the machine M_r the machine stops after at most 2^r computational steps, *whether or not* it has finished testing $\mathcal{P}_n$. Since M_r has clock time 8^{-r} times the clock time of M_1, this means that it does not spend longer than a time 4^{-r} measured in the clock time of M_1. If after this time P_n is not resolved, the machine constructs the machine M_{r+1} and passes the problem on WITHOUT increasing the value of n. The next machine proceeds in the same way. Eventually somewhere down the line the problem is resolved positively or negatively for that value of n, and one can pass to the case $n+1$ or report success up the chain. Since every M_r takes no more time on the problem than 4^{-r}, the overall total time to complete the task is finite.

If there is a solution to the problem then one of the machines will find it and pass the news back to the first machine. However, it may well be that the smallest solution is so large that it cannot be stored within M_1. All that we can be sure of storing in M_1 is the fact that the solution has been found. It would, however, be possible for M_1 to print out the entire solution, if the digits are transmitted up the chain in batches which are small enough for M_1 to be able to accept them into memory for printing and then delete them from memory before taking the next batch. It may not be possible either to print out the smallest solution in less than, say, a trillion years, nor even to indicate how many digits it has within such a time.

We next discuss in more detail how a report is passed up the chain, assuming that it consists of a single bit. Longer reports can be dealt with similarly. We assume that the clocks are synchronised so that the start T of any clock period of M_1 is also the start of a clock period of all the other machines. Let the machine M_n pass a bit up to M_{n-1} during the first clock period of M_n following T; remember that the clock times of M_{n-1} are 8 times longer than those of M_n. Allowing for a finite passage time this may arrive during the second clock period of M_{n-1} after T. M_{n-1} may then transfer the bit to its output device during its third clock period, and transmit it to M_{n-2} during its fourth clock period. This is still in the middle of the first clock

period of M_{n-2} so it should arrive during the second clock period of M_{n-2}. We are now in an inductively stable situation so the bit eventually arrives at M_1 in its second clock period after T.

So far we have only shown how to solve a problem of the type: Does there exist n for which $\mathcal{P}_n$ is true? A more difficult case is the problem: Do there exist infinitely many n for which $\mathcal{P}_n$ is true? A particular case is the existence of infinitely many prime pairs. Our machine can also solve this problem as follows.

We start by putting $m = 1$. We ask M_1 to find out whether there exists $n > m$ for which $\mathcal{P}_n$ is true, by the procedure just described. If the answer is yes, it does not report to the outside world as before, but passes the same problem to M_2 together with the number $m = 2$. Inductively if M_n receives the answer YES it passes the problem to M_{n+1} with $m = n + 1$. If at any stage the answer is NO, then the report NO is passed back up the chain all the way to M_1. So in this case the absence of a report within a suitable finite length of time proves that an infinite number of solutions do exist and the presence of a report proves that only a finite number of solutions exist. For the same reason as before, it may not be possible for M_1 to communicate the values of the largest number n for which $\mathcal{P}_n$ is true in less than a trillion years. The fact that one of the machines down the line already possesses the value is of little comfort.

We next discuss Turing's Halting Problem, that of deciding whether a particular program runs for ever on a Turing machine or eventually stops. Turing proved that there is no algorithm for identifying the programs which run for ever. If one is allowed to use our machines, which are not algorithmic in the strict sense of the term, then such a procedure does exist. Given the program, starting with the case $n = 1$ we run it on M_n until either it has stopped, or the memory of M_n is full, or M_n has carried out 2^n steps, whichever happens first. If at that time the program has not stopped we pass the same program to M_{n+1}. Although each machine runs the program for twice as many steps as the previous one the time taken decreases geometrically, because each machine runs faster. If the program stops on any of the machines then a report is sent back to M_1. If M_1 receives no report after a certain finite length of time then it can conclude that the program would run for ever on a Turing machine.

We next show that when carrying out the above tasks our machines do not suffer from what is called the Thompson lamp paradox, the possibility that after completing an infinite computation they end up in an indeterminate state (Earman and Norton [1998]; Hogarth [1996]; Thompson [1954–55]). At the end of the task set, each machine has taken the program and carried out some computations, after which it either reports success upwards or passes the program to a new and smaller machine which it builds. The program may

include the instruction that once it has been run on any machine M_n that machine is to clear its memory. Each machine may also have built into its hardware the instruction to pass on any message received from machines down the chain without alteration. In this case every machine will be in a neutral state when the computation is completed, whether or not the problem is soluble.

Nevertheless, there was only one machine at the start while there are an infinite number (with a finite total amount of material) at the end, if the problem is insoluble. The first machine may be given the extra command that it should detach the second machine when it knows that the computation is finished, and return the material from the remainder to its store of building material. This guarantees that the final situation is the same as the initial situation: one machine with empty memory.

In spite of the above, problems might arise if M_1 is given a program to run which is not of the above two types. It seems possible, at least in principle, that one might get into a state in which every machine in the chain is trying to send and to receive information from both of the two machines next to it in the chain. We have not specified the communication protocols sufficiently precisely to ensure that a blockage cannot occur. Such problems are routine on the internet, which also has to cope with machines running at different speeds, but only a finite number of them.

Each machine has five basic modes: computing, sending information up or down the chain, and receiving information from up or down the chain. If we refer to these as (c), (su), (sd), (ru) and (rd), then it seems that the chances of a blocked state or of memory overflow are minimized by giving these the priorities (c), (su), (ru), (rd), (sd). We also specify that once a program has been completed or data transmitted it should be cleared from memory. Data should be sent in packets of a size which is acceptable by the machine with the smallest memory, namely M_1. Finally data should be transmitted at the highest rate which both of the machines involved can accept. We conjecture that with such protocols the system does not get stuck in some strange state when trying to solve a genuine mathematical problem.

4 Physical issues

One of the problems of a continuous Newtonian universe which did not disturb anyone until the twentieth century is that it is not clear that solid matter can exist in it. In our own universe the Lieb-Thirring proof of the stability of matter depends upon quantum mechanics and the Pauli exclusion principle. Moreover, the formulation of the problem presupposes that matter is atomic. It is entirely unclear what might hold solid matter together in our universe. We consider that such problems are not capable of intelligent

discussion, since the structure of matter in our own universe has taken hundreds of years to understand even with the advantage of experimental investigation. At some point one must accept that thought experiments have only limited validity.

An important difference between the universe which we have been describing and the real one relates to thermal noise and information theory. There is a large literature on such matters for which we refer to Barrow and Tipler ([1986]), Beckenstein ([1981]), Bennett ([1984]), and Porod, Grondin and Ferry ([1984]) and sources cited there. The most obvious problem is that our machine is in conflict with the time-energy uncertainty principle, which forces signals that are transmitted ever more rapidly to become more energetic without limit. But this uncertainty principle involves Planck's constant, which vanishes if the universe is purely classical. Wigner's lower bounds on the size and mass of a clock running at a given speed are not relevant for similar reasons (Barrow [1996]).

Our machines are required to transmit the program without error from one to the next more and more rapidly as one passes down the hierarchy. To achieve this it is simplest to assume that the transmission channels are noiseless. Each of the machines M_n only interacts with the outside world via the previous one, and we see no fundamental reason why it should not be a reversible computation device, since Newton's laws are time-reversible. This might be hard to achieve and would certainly entail the avoidance of spring-loaded rachets. Only one bit of information need be sent up the chain to the first machine and hence to the outside world, so it might be an oracle machine. Such a machine lives in a highly idealized universe and might be said to run at zero temperature.

It is not possible to describe the effects of noise on our machines without making some assumptions about its origin. In particular it is difficult to discuss thermal noise, since our understanding of this in the real world depends upon an atomic model—statistical mechanics. Suppose instead that the machine M_n breaks down with probability p_n because of defective teeth on cog wheels. If the failures of the machines are statistically independent and p_n does not depend on n, then the total system is certain to fail. However, the above assumptions may not be appropriate. If the failure is due to impurities in the raw material then the successful operation of the first n machines may indicate that the particular sample of raw material used is very pure and hence may lead to a very much higher probability that the later machines will also work as intended. In other words, if the probabilities of failure of the different machines are highly correlated, then it is quite possible that the total system will work as intended with high probability.

If breakdowns are caused by the cog wheels wearing out, then one cannot determine whether the machine will function as intended without

assumptions on the failure rate. If one assumes that the failure rate is the same per complete rotation for every wheel then the machine must fail before completing an infinite computation. However, the velocity of components does not increase as their size decreases, so it is plausible that the failure rate would be constant per unit time. In this case the total system may work as intended with high probability.

Before continuing, we should mention that physical realizations of Turing machines have several of the defects of the machines discussed in this paper, but this has not prevented the appearance of a vast literature discussing their logical significance. Turing machines have infinite memories, and could not be built in a finite spacetime. An infinite, uniform, one-dimensional memory tape in a Newtonian universe has infinite mass and an infinite gravitational self-energy. This indicates that an *extremely* large finite memory tape cannot be constructed in the real world. A Turing machine is assumed to operate without errors, and would fail instantly in a quantum universe because of quantum fluctuations somewhere along its infinite length. A Turing machine is stochastically unstable: an infinitely long tape placed in a Newtonian universe containing a very sparse uniformly but randomly distributed collection of very small moving bodies would be destroyed instantly by multiple collisions at some remote location. Similar considerations apply if the memory cells have a very small probability of defects, independently for each cell. Discussions about the possibility of a Turing machine embarking on a sufficiently long *finite* computation ignore the following fact: we have no reason to believe that the real universe will retain its present characteristics long enough for more than 10^{1000} steps of any computation on any conceivable machine to be performed. For these reasons we consider that the rejection of *our* machines on the grounds of physical implausibility should imply the same attitude towards Turing machines.

5 Philosophical implications

Neither our machines nor Turing machines can actually be built, because of fundamental properties of the real universe. On the other hand both are plausible within a sufficiently constrained universe. Both are idealizations of finite computing machines, and both are constructed by repeatedly adding on extra stages to a finite machine. This section discusses whether such thought experiments have any philosophical implications.

Turing machines were introduced by Turing in 1936 in order to discuss mathematical issues, and only later did their partial physical implementation by electronic computing machines become important. Much of what he wrote about mechanical processes implemented by computers referred to repetitive procedures carried out following fixed rules by human beings at a time when

being a computer was a job. But the fact that he was considering human computation rather than machine computation does not affect any of the fundamental problems. Human beings cannot carry out sufficiently large finite computations because they need to store intermediate results on paper or some other medium. For a large enough problem the resources of the universe will not be sufficient, even for the 'idealized' human who lives for ever. In other words there are two notions of proof, one purely logical and analytic, which does not take account of whether the computations can be completed in the real world, and the other physical and empirical, which does. Turing and most of his successors did not make this distinction.

The possibility of constructing our machines in a suitable universe does not disprove Gödel's theorem even though the Halting Problem changes its form if one is allowed to use our machines. We refer to Hogarth [(1996)] for a discussion of these issues and to Earman and Norton [(1996)] for a discussion of several other philosophical questions which such machines raise. The issue to which our machines might be considered most relevant is the Platonist-Intuitionist debate about the number system. One might argue that in a universe in which our machine could be constructed, the finitist scruples of Weyl and Brouwer would be less justified. At least some questions about the entire set of numbers could be resolved in a finite time by an infinite brute-force search, so it would be less easy to argue that the set of all integers does not exist as a completed entity. (I am fully aware that most mathematicians behave as if this debate has long been settled in favour of the existence of the completed set of integers, as well as of many other much more abstract sets. However, the philosophical debate is still alive, for very good reasons.)

On the other hand, our universe is not a continuous Newtonian universe, and we have only given an outline of how to construct such a machine. The Newtonian universe only exists in our imagination, so the finite time solutions to the infinite problems only exist in our imagination. To claim that the existence of a solution to some infinite problem about numbers in an imagined universe implies *anything* about the mathematics of our universe depends upon a philosophical assumption. This is that the imagined mathematics of the imagined universe has the same properties as the real mathematics of ours. This is a form of mathematical Platonism and will not convince someone who believes that numbers are social constructions. I quote Bishop [(1967)].

> We are not interested in properties of the positive integers which have no descriptive meaning for finite man. When a man proves a positive integer to exist, he should show how to find it. If God has mathematics of his own that needs to be done, let him do it himself.

It seems that Platonists will consider that the construction of our machine only provides further support for what they already believed about the

objective status of the completed set of integers. On the other hand Intuitionists will see yet another illustration of how easy it is to be misled by fantasies which are unrelated to what can actually be proved in the real world. At the very least it is interesting how detailed a discussion one can have about the properties of a non-existent universe.

It should be mentioned that modal realists claim that possible worlds are just as real as the one in which we live, but are parts of reality with which we have no contact; see Lewis ([1986]) and Stalnaker ([1996]) for accounts of this theory. From such a point of view it might be argued that any mathematical result which can be demonstrated in one of the worlds is then automatically true in all the others, even if it is not provable in them. To use this argument in the present context one would have to establish that the text of this paper does describe a possible world. In the author's opinion the text is no more than a thought experiment, and a continuous Newtonian world might not be possible except as an idea in the mind of a person who has already accepted the current body of mathematics. I leave such matters for others to discuss.

There is another issue to which the existence of such a computer may be relevant. Some mathematicians consider that their goal is to attain understanding, not just knowledge. To be told that the prime pair hypothesis is true simply because a computer has found an infinite number of prime pairs would be highly unsatisfactory. Maybe this is the best that can be done: the fact is a contingent reality and no finite proof exists. But no mathematician would believe it, and the record of other hypotheses suggests that he/she would be right to continue looking for a proof. Of course an infinite computer might be able to run through all formal arguments until it has found a finite proof of the theorem or reported back that none existed. And it might be asked to send back the shortest proof if it was within the capacity of the first machine to accept it. Mathematics in such a universe would be totally unlike ours—if such a universe existed.

Acknowledgements

We would like to thank J. D. Barrow, M. Hogarth, L. J. Landau, J. P. Laraudogoitia, J. D. Norton, R. F. Streater and the referees for helpful comments.

Department of Mathematics
King's College London
Strand
London WC2R 2LS
UK
E.Brian.Davies@kcl.ac.uk

References

Barrow, J. D. and Tipler, F. J. [1986]: Chapter 10 of 'The Anthropic Cosmological Principle', Oxford: Oxford University Press.

Barrow, J. D. [1996]: 'Wigner inequalities for a black hole', *Phys Rev* **54D**, pp. 6563–4.

Beckenstein, J. D. [1981]: 'Energy cost of information transfer', *Phys. Rev. Lett.* **46**, pp. 623–6.

Bennett, C. H. [1984]: 'Thermodynamically reversible computation', *Phys. Rev. Lett.* **53**, p. 1202.

Bishop, E. [1967]: *Foundations of Constructive Analysis*, New York: McGraw-Hill.

Boolos, G. S. and Jeffrey, R. C. [1989]: *Computability and Logic*, Cambridge: Cambridge University Press.

Earman, J. and Norton, J. D. [1996]: 'Infinite Pains: The Trouble with Supertasks' in A. Morton and S. P. Stich (*eds*), 1996, *Benacerraf and his Critics*, Cambridge, MA: Blackwell, pp. 231–61.

Earman, J. and Norton, J. D. [1998]: 'Discussion: Comments on Lauradogoitia's ''classical particle dynamics, indeterminism and a supertask'' ', *The British Journal for the Philosophy of Science*, **49**, pp. 123–33.

Hogarth, M. [June 1996]: 'Predictability, Computability, and Spacetime', PhD thesis, Cambridge University.

Lewis, D. [1986]: *On the Plurality of Worlds*, Oxford: Blackwell.

Porod, W., Grondin, R. O. and Ferry, D. K. [1984]: 'Dissipation in computation', *Phys. Rev. Lett.* **52**, pp. 232–5.

Stalnaker, R. [1996]: 'On what possible worlds could not be' in A. Morton and S. P. Stich (*eds*), 1996, *Benacerraf and his Critics*, Cambridge, MA: Blackwell, pp. 103–18.

Thompson, J. [1954–55]: 'Tasks and supertasks', *Analysis*, **15**, pp. 1–13.