Constraints on hypercomputation

Greg Michaelson[1] and Paul Cockshott[2]

[1] Heriot Watt University
G.Michaelson@hw.ac.uk
[2] University of Glasgow
wpc@dcs.gla.ac.uk

Abstract. Wegner and Eberbach[16] have argued that there are fundamental limitations to Turing Machines as a foundation of computability and that these can be overcome by so-called superTuring models. In this paper we contest their claims for interaction machines and the πcalculus.

1 Introduction

The Turing machine (TM) [1] has been the dominant paradigm for Computer Science for 70 years: Ekdahl[12] likens an attack on it to a "challenge to the second law of thermodynamics".

The roots of Turing's work lie in debates about the notion of computability in the pre-computer age[10]. Just before the Second World War, in an outstanding period of serendipity, Turing, Church and Kleene all developed independent notions of computability which were quickly demonstrated to be formally equivalent. These seminal results form the basis for the Church-Turing Thesis that all notions of computability will be equivalent. Until now, the Church-Turing Thesis has remained unshaken.

A central concern of these pre-computer Mathematical Logicians was to formalise precisely the concept of effective computation. For Church, this is a matter of *definition*, explicitly identifying effective calculability with recursive or lambda-definable functions over the positive integers. Church[5] states that:

> If this interpretation or some similar one is not allowed it is difficult to see how the notion of an algorithm can be given any exact meaning at all.(p356)

Turing subsequently outlined a proof of the equivalence of his notion of "computability" with Church's "effective calculability".

A fundamental distinction of Turing's approach is that he identifies a human-independent mechanism to embody his procedure, by explicit analogy with a human being:

> We may compare a man in the process of computing a real number to a machine which is only capable of a finite number of conditions... ([1]Section 1).

In a 1939 paper discussing the unity of these different approaches, Turing is explicit about the mechanical nature of effective calculation:

> A function is said to be "effectively calculable" if its values can be found by some purely mechanical process ... We may take this statement literally, understanding by a purely mechanical process one which may be carried out by a machine. It is possible to give a mathematical description, in a certain normal form, of the structures of these machines. ([2]p166).

In a late paper he makes the same point with reference to digital computers:

> The idea behind digital computers may be explained by saying that these machines are intended to carry out any operations which could be done by a human computer. ([3]Section 4).

For Church and Kleene the presence of a human mathematician that applies the rules seems to be implicit, but the ability to give an explicit procedure for applying rules to symbols and to physically realise these procedures, was central to all three conceptions of effectiveness. The corollary is that a computation which is not physically realisable is not effective.

2 Wegner and Eberbach's SuperTuring Computers

There has been robust debate in Mathematics, Philosophy, Physics and, latterly, Computer Science about the possible of *hypercomputation* which seeks to transcend the limits of classic computability. Copeland [6] and Cotogno[8] provide useful summaries.

Thus, Wegner and Eberbach[16] assert that the fundamental limitations to the paradigmatic conception of computation can be overcome by more recent "superTuring" approaches. They draw heavily on the idea of an algorithm as an essentially closed activity. That is, while the TM realising an algorithm may manipulate an unbounded memory, the initial memory configuration is pre-given and may only be changed by the action of the machine itself. Furthermore, an effective computation may only consume a finite amount of the unbounded memory and of time, the implication being that an algorithm must terminate to be effective.

They say that the TM model is too weak to describe the Internet, evolution or robotics. For the Internet, web clients initiate interactions with servers without any knowledge of the server history.

Wegner and Eberbach claim that there is a class of superTuring computations (sTC) which are a superset of TM computations. That is sTC includes computations which are not realisable by a TM. A superTuring computer is "any system or device which can carry out superTuring computation". They give discursive presentations of interaction machines (IM), the π-calculus and the \$-calculus, and explore why they transcend the TM. here, we do not consider the \$-calculus as its sTC properties appear to depend on those alleged for interaction machines and π-calculus.

3 How might the TM paradigm be displaced?

In general, a demonstration that a new system is more powerful than a C-T system involves showing that while all terms of some C-T system can be reduced to terms of the new system, there are terms of the new system which cannot be reduced to terms of that C-T systemMore concretely, we think that requirements for a new system to conclusively transcend C-T are, in increasing order of strength:

1. demonstration that some problem known to be semi-decidable in a C-T system is decidable in the new system;
2. demonstration that some problem known to be undecidable in a C-T system is semi-decidable in the new system;
3. demonstration that some problem known to be undecidable in a C-T system is decidable in the new system;
4. characterisations of classes of problems corresponding to 1-3;
5. canonical exemplars for classes of problems corresponding to 1-3.

Above all, we require that the new system actually encompasses effective computation; that is, that it can be physically realised in some concrete machine. While we are not unduly troubled by systems that require potentially unbounded resources such as an unlimited TM tape, we reject systems whose material realisations conflict with the laws of physics, or which require actualised infinities as steps in the calculation process.

4 Physical Realism and Computation

A key point about the Universal Computers proposed by Turing is that they are material apparatuses which operate by finite means. Turing assumes that the computable numbers are those that are computable by finite machines, and initially justifies this only by saying that the memory of a human computer is necessarily limited.

Turing is careful to construct his machine descriptions in such a way as to ensure that the machine operates entirely by finite means and uses no techniques that are physically implausible. His basic proposition remained that :"computable numbers may be described briefly as the real numbers whose expressions as a decimal are calculable by finite means."

Turing rules out computation by infinite means as a serious proposition. Most proposals for superTuring computation rest on the appeal of the infinite. Copeland[6] proposes the idea of accelerating Turing machines whose operation rate increases exponentially so that if the first operation were performed in a microsecond, the next would be done in $\frac{1}{2}\mu s$, the third in $\frac{1}{4}\mu s$, etc. The result would be that within a finite interval it would be able to perform an infinite number of steps. This evades all possibility of physical realisation. A computing machine must transfer information between its component parts in order to perform an operation. If the time for each operation is repeatedly halved. then one soon reaches the point at which signals traveling at the speed of light have insufficient time to propagate from one part to another within an operation step. Hamkins[14] discusses what could be computed on Turing machines if they were allowed to operate for an infinite time. He hypothesises a relativistic experiment in which a researcher moving near the speed of light experiences a finite duration whilst back on Earth his graduate student has an infinite duration to solve a problem. Hamkins fails to suggest which immortal graduate student he has in mind for this task. Another theme of those advocating Super-Turing computation is the use of analogue computation over real numbers. For a review see [7]. The idea of being able to physically represent real numbers is highly questionable in view of the quantum of action h. This poses fundamental and finite limits on the accuracy with which a physical system can approximate real numbers.

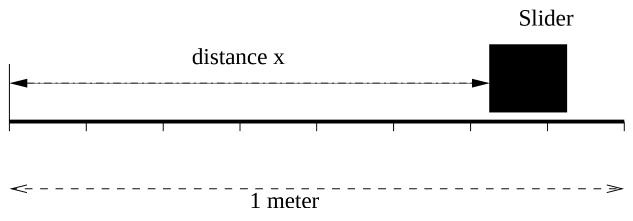

Fig. 1. An analogue representation of a real number using a physical version of the real number line.

Suppose we want to use analogue encoding real numbers as spatial separation, as shown in Figure 1,to encode a real number x that could be used by some Super Turing analogue computational process. We first set up the the distance x and then measure it during the process of the computation.This raises two questions:

1. How precisely can we, in principle at least, measure the distance x?
2. How stable is such an analogue memory. For long can it store the information.

Because of Heisenberg's equation

$$\Delta p \Delta x = \frac{h}{4\pi}$$

there is a tradeoff between the accuracy to which x can be represented as distance and the period for which x can be stored. If the mass of the slider is m the uncertainty in its velocity is given by $\Delta v = \frac{h}{4\pi m \Delta x}$. This implies that the persistence time of the analogue store T_p is constrained such that

$$T_p \approx \frac{\Delta x}{\Delta v} = \frac{4\pi m \Delta x^2}{h}$$

It is clear that for a practical device, T_p must be chosen exceed the time taken to measure the distance x. This in turn, must be greater than $\frac{2x}{c}$, since any distance measurement will involve at least one reflection off the slider. If $T_p < \frac{2x}{c}$ then the computer would be unable to read the real. We thus have the constraint

$$\frac{2x}{c} < \frac{4\pi m \Delta x^2}{h}$$

which implies

$$\Delta x^2 > \frac{2xh}{4\pi mc}$$

For a 1 kilo slider adjustable over a range of 1 meter which allows the representation of reals in the range 0..1 as $x \pm \Delta x$, we find that $\Delta x > 5.93 \times 10^{-22}$meters.

This corresponds to a real number stored with about 70 bits of precision.

To add an additional bit of precision to our real number we would have to quadruple the mass of the slider. The analogue encoding of the reals requires a mass which grows with $\mathbf{O}e^{2b}$ where b is the number of bits of precision to which the reals are stored. For a Turing Machine the mass required grows linearly with the number of bits. Thus proposals to incorporate the full mathematical abstraction of real numbers into computing devices so as to allow them to outperform Turing machines are physically implausible.

5 Interaction Machines

Wegner and Eberbach claim that a Turing Machine is restricted by having to have all its inputs appear on the tape prior to the start of computation. Interaction machines on the contrary can perform input output operations to the environment in which they are situated. Interaction Machines, whose canonical model is the Persistent Turing Machine(PTM) of Goldin [9], are not limited to a pre-given finite input tape, but can handle potentially infinite input streams. These arguments have been thoroughly criticised by Ekdahl[12]. Rather than rehearse his arguments we shall focus on additional weaknesses of Wegner and Eberbach's claims.

5.1 Turing's own views

Turing's Test for machine intelligence is probably as well known as his original proposal for the Universal Computer. He proposed in a very readable paper[3], that a computer could be considered intelligent if it could fool a human observer into thinking they were interacting with another human being. It is clear that his putative intelligent machine would be an Interaction Machine in Wegner's sense. Rather than being cut off from the environment and working on a fixed tape, it receives typed input and sends printed output to a person.

Turing did notfind it necessary to introduce a fundamental new class of computing machine for this gedanken experiment. He describes the machine using what is a paraphrase (Turing 1950, page 436) of his description of the computing machine in his 1936 paper. It is clear that Turing is talking about the same general category of machine in 1950 as he had in 1936. He says he is concerned with discrete state machines, and that a special property of such digital computers was their universality:

> This special property of digital computers, that they can mimic any discrete state machine, is described by saying that they are universal machines. The existence of machines with this property has the important consequence that, considerations of speed apart, it is unnecessary to design various new machines to do various computing processes. They can all be done with one digital computer, suitably programmed for each case. It will be seen that as a consequence of this all digital computers are in a sense equivalent.(Turing 1950, page 442)

This is clearly a recapitulation of the argument in section 6 of his 1936 paper where he introduced the idea of the Universal Computer. Turing argued that such machines were capable of learning and that with a suitable small generalised learning program and enough teaching, then the computer would attain artificial intelligence.

5.2 Equivalence of Interaction Machines and Turing Machines

Consider first a digital computer interacting in the manner forseen by Turing in his 1950 paper, with teletype input/output. Suppose then we have a computer initialised with a simple learning program following which it is acquires more sophisticated behaviour as a result of being 'taught'. As the computer is taught we record every keystroke onto paper tape.

We initialise a second identical computer with the same program' and at the end of the first computer's working life we give to the second machine as an input, the tape on which we have recorded the all the data fed to the first machine. With the input channel of the second machine connected to the tape reader it then evolves through the same set of states and produce the same outputs as the original machine did. The difference between interactive input from a teletype and tape inputis essentially trivial.

A small modification to the program of a conventional TM will transform it into a PTM. Like Goldin we will assume a 3 tape TM, M_1 with one tape T_1 purely for input, one tape T_2 purely for output and one tape T_3 used for working calculations. We assume that tapes T_1, T_2 are unidirectional, T_3 is bidirectional. M_1 has a distinguished start state S_0 and a halt state S_h. On being set to work it either goes into some non-terminating computation or eventually outputs a distinguished termination symbol τ to T_2, branches to state S_h and stops. We assume that all branches to S_h are from a state that outputs τ. Once τ has been output, the sequence of characters on T_2 up to to τ are the number computed by the machine.

We now construct a new machine M_2 from M_1 as follows: replace all branches to S_h with branches to S_0. From here it will start reading in further characters from T_1 and may again evolve to a state where it outputs a further τ on T_2. Machine M_2now behaves as one of Goldin's PTMs. It has available to it the persisting results of previous computation on $T_3$and these results will condition subsequent computations. It is still a classic TM, but a non-terminating one. It follows that PTM's, and thus Interaction Machines of which they are the canonical example, are a sub-class of TM programs and do not represent a new model of computation.

6 π-Calculus

The π-calculus is not a model of computation in the same sense as the TM: there is a difference in level. The TM is a specification of a material apparatus that is buildable. Calculi are rules for the manipulation of strings of symbols and these rules will not do any calculations unless there is some material apparatus to interpret them.

Is there any possible physical apparatus that can implement the π-calculus and, if so, is a conventional computer such an apparatus. Since it is possible to write a conventional computer program that will apply the formal term re-write rules of the π-calculus to strings of characters representing terms in the calculus [15] then it would appear that the π-calculus can have no greater computational power than the von Neumann computer.A possible source of confusion is the language used to describe the π-calculus: channels, processes, evolution which imply that one is talking about physically separate but communicating entities evolving in space/time. There is a linguistic tension between what is strictly laid down as the rules of a calculus and the rather less specific physical system that is suggested by the language. One has to be very careful before accepting that the existence the π-calculus as a formal system implies a physically realisable distributed computing apparatus.

Consider two of the primitives: synchronisation and mobile channels.

Is π-calculus synchronisation in its general sense physically realistic?

Does it not imply the instantaneous transmission of information - faster than light communication if the processes are physically separated?

If the processors are in relative motion, there can be no unambiguous synchronisation shared by the different moving processes. It thus follows that the processors can not be physically mobile if they are to be synchronised with at least 3 way synchronisation (see [11] pp 25-26).

Suppose we have the following pi calculus terms

$$\alpha \equiv (\bar{a}v.Q) + (by.R[y]) \tag{1}$$

$$\beta \equiv (\bar{b}z.S) + (ax.T[x]) \tag{2}$$

In the above α and β are processes. The process αtries to either output the value v on channel a or to read from channel b into the variable y. The $+$ operator means non deterministic composition, so $A + B$ means that either A occurs or B occurs but not both. The notation $\bar{a}v$ means output v to a, whilst av would mean input from a into v. If α succeeds in doing an output on channel a it then evolves into the abstract process Q, if alternatively, it succeeds in doing an input from b into y, then it evolves into the process $R[y]$ which uses the value $y$in some further computation.

We can place the two processes in parallel :

$$(\bar{a}v.Q) + (by.R[y])|(\bar{b}z.S) + (ax.T[x]) \tag{3}$$

This should now evolve to

$$(Q|T[v]) \textbf{ or to } (S|R[z]) \tag{4}$$

where either Q runs in parallel with $T[v]$ after the communication on channel a or where S runs in parallel with $R[z]$ after the value z was transfered along channel b from process $\beta$to process α.

Since the two processes are identical mirror images of one another any deterministic local rule by which process β commits to communication on one of the channels, must cause α to commit to the other channel and hence synchronisation must fail.

Thus if α commits to communication on channel a then its mirror image β must commit to communicate on b leading to: $T[x]|R[y]$, but this is not permitted according to the π-calculus.

The argument is a variant of the Liar Paradox, but it is not a paradox within the π-calculus itself. It only emerges as a paradox once you introduce the constraints of relativity theory prohibiting the instantaneous propagation of information. Nor does abandoning determinism help. If the commitment process is non-deterministic, then on some occasions synchronisation will succeed, but on other occasions the evolution both processes will follow the same rule, in which case synchronisation will fail.

A global arbitration machine solves the problem at the loss of parallelism. A worse loss of parallelism, in terms of complexity order, is entailed by distributed broadcast protocols such as Asynchronous Byzantine Agreement[4].

In conclusion it is not possible to build a reliable mechanism that will implement in a parallel distributed fashion any arbitrary composition of π-calculus processes.

6.1 Wegner and Eberbach's argument

Wegner's argument for the super-Turing capacity of the π-calculus rests on there being an implied infinity of channels and an implied infinity of processes. Taking into account the restrictions on physical communications channels the implied infinity could only be realised if one had an actual infinity of fixed link computers. At this point we are in the same situation as the Turing machine tape - a finite but unbounded resource. For any actual calculation a finite resource is used, but the size of this is not specified in advance.W&E then interprets 'as many times as is needed' in the definition of replication in the calculus as meaning an actual infinity of replication. From this he deduces that the calculus could implement infinite arrays of cellular automata for which he cites Garzon [13] to the effect that they are more powerful than TMs.

7 Conclusion

In Section 3, we gave criteria that must be met for the Church-Turing thesis to be displaced. In general, a demonstration that all terms in C-T systems should have equivalent terms in the new system but there should be terms in the new system which do not have equivalents in C-T systems. In particular, the new system should be able to solve decision problems that are semi-decidable or undecidable in C-T systems. Finally, we require that a new system be physically realisable. We think that, under these criteria, Wegner and Eberbach's claims that Interaction Machines, the π-calculus and the \$-calculus are super-Turing are not adequately substantiated.

First of all, Wegner and Eberbach do not present a concrete instance of terms in any of these three systems which do not have equivalents in C-T systems. Secondly, they do not identify decision problems which are decidable or semi decidable in any of these systems but semi-decidable or undecidable respectively in C-T systems. Finally, they do not explain how an arbitrary term of any of these three systems may be embodied in a physical realisation. We have shown that the synchronisation primitive of the calculus is not physically realistic. The modeling of cellular automata in the calculus rests on this primitive. Furthermore, the assumption of an infinite number of processes implies an infinity of mobile channels, which are also unimplementable. We therefore conclude that whilst the π-calculus can be practically implemented on a single computer, infinite distributed implementations of the sort that W&E rely upon for their argument can not be implemented.

Acknowledgments

Greg Michaelson wishes to thank the School of Information Technology at the University of Technology, Sydney for their hospitality during the writing of this paper.

References

1. A. M. Turing. On Computable Numbers with an Application to the Entschiedungsproblem. *Proc. London Mathematical Soc.*, 42:230–265, 1936.
2. A. M. Turing. Systems of Logic Based on Ordinals. *Proc. London Mathematical Soc.*, 45, 1939.
3. A. M. Turing. Computing Machinery and Intelligence. *Mind*, 39:433–60, 1950.
4. G. Bracha and S. Toueg. Asynchronous consensus and byzantine protocol in a faulty environment. Technical Report TR-83-559, CS Dept., Cornell University, Ithaca,NY 14853, 1983.

5. A. Church. An Unsolvable Problem of Elementary Number Theory. *American Journal of Mathematics*, 58:345–363, 1936.
6. B. J. Copeland. Hypercomputation. *Minds and Machines*, 12:461–502, 2002.
7. B. J. Copeland and R. Sylvan. Beyond the universal turing machine. *Australasian Journal of Philosophy*, 77(1):46..66, 1999.
8. Paolo Cotogno. Hypercomputation and the Physical Church-Turing Thesis. *Brit. J. Phil. Sci.*, 54:181–223, 2003.
9. D. Goldin, S. Smolka and P.Wegner. Turing Machines, Transition Systems, and Interaction. *Information and Computation*, 194(2):101–128, November 2004.
10. Martin Davis. *Engines of Logic : Mathematicians and the Origins of the Computer.* Norton, 2001.
11. A. Einstein. *Relativity.* Methuen and Company, London, 1920.
12. B. Ekdahl. Interactive Computing does not supersede Church's thesis. In *Proc. Computer Science*, pages 261–265, 17th Int. Conf. San Diego, 1999. Association of Management and the International Association of Management,.
13. M Garzon. *Models of Massive Parallelism: Analysis of Cellular Automata and Neural Networks.* EATCS. Springer-Verlag, 1995.
14. J. Hamkins and A. Lewis. Infinite Time Turing Machines. *Journal of Symbolic Logic*, 65(2):567–604, 2000.
15. D. Turner and B. Pierce. Pict: A programming language based on the picalculus. In G. Plotkin, C. Stirling, and M. Tofte, editors, *Proof, Language and Interaction: Essays in Honour of Robin Milner*, pages 455–494. MIT Press, 2000.
16. P. Wegner and E. Eberbach. New models of computation. *Computer Journal*, 47:4–9, 2004.