Non-classical computing: feasible versus infeasible.

Paul Cockshott, Lewis Mackenzie
Department of Computer Science
University of Glasgow
{wpc,lewis}@dcs.gla.ac.uk

Greg Michaelson
School of Mathematical and Computer Sciences
Heriot-Watt University
G.Michaelson@hw.ac.uk

Physics sets certain limits on what is and is not computable. These limits are very far from having been reached by current technologies. Whilst proposals for hypercomputation are almost certainly infeasible, there are a number of non classical approaches that do hold considerable promise. There are a range of possible architectures that could be implemented on silicon that are distinctly different from the von Neumann model. Beyond this, quantum simulators, which are the quantum equivalent of analogue computers, may be constructable in the near future.

The Infinite Improbability Drive is a wonderful new method of crossing vast interstellar distances in a mere nothingth of a second, without all that tedious mucking about in hyperspace.Adams & Stamp (1997)

1. INTRODUCTION

At the end of the first decade of the 21st century, Computing has the hallmark of a mature discipline, providing the intellectual and pedagogical foundations for the key technology of just about all industrial, commercial and social activity across our peculiar planet. Computers truly are ubiquitous and pervasive: there are probably more CPUs in a typical G20 household than in the whole of the world mid-20th century. Quite apart from personal computers, laptops, PDAs and smart phones, general purpose processors are found in watches, white goods, televisions, media players, games, toys and so on. All bear testament to the enormous theoretical and practical legacy bequeathed to us by the pioneering mathematicians and engineers of that brief, yet astonishingly fertile, period spanning the Second World War.

The mid-1930's saw the serendipitous elaborations of Turing's machines, Church's λ calculus and Kleene's recursive function theory. We can directly trace our current formalisations of computing systems back to these roots. Thus Turing machines underpin contemporary information and compexity theory, while λ calculus and recursive function theory underpin language semantics, type systems, process calculi, cost modelling, transformation, refinement, theorem proving and automated reasoning. Indeed, the Church-Turing thesis, that all systems purporting to capture common sense notions of universal computation are mutually equivalent, is validated with each new candidate.

Practical computing machines have even older roots. The vision of Babbage's analytic engine was clouded by the technological limitations of its era. Nonetheless, we can see all the abstractions of contemporary computers in its imaginings. The heroic elaborations of mechanical analogue computers, most effectively realised in wartime naval and anti-aircraft gunnery systems, again hit technological limitations with large scale differential analysers. Post-war, after brief competition from electronic based analogue computers, our familiar digital computer emerged from the designs of von Neumann and his contemporaries.

And the Church-Turing thesis applies also to physical realisations of computing machines, allowing for potentially infinite memory. In turn, the thesis applies to the languages, or more properly to the abstract machines, with which we construct computing systems. In particular, all the limitations identified by the undecidability results for classic Church-Turing systems apply to our software systems as well. From Turing's initial proof that the Halting Problem for Turing machines is undecidable, it has been subsequently established that it is impossible to tell if two arbitrary Turing machines are equivalent, or how much time and memory a Turing machine will require to complete a computation on an arbitrary initial tape.

When applied to our systems, the practical implications are stark. Thus, in general, we cannot tell:

- if one piece of software is a plagiarism of another;
- whether a computer has enough memory to solve some instance of a problem;
- if formally tractable problem will complete in a time we consider psychologically acceptable;
- for an arbitray pattern of use, how long the batteries on a mobile device will last for.

Of course these limitations really do not impinge directly on most everyday deployment and use of computers[1]. By analogy with Physics, it would be nice to travel faster than light or to go back in time or to exploit perpetual motion, but we are generally comfortable accepting these long established constraints.

Or are we? Every year, people, frustrated that they cannot realise science fiction dreams, claim to have built devices that can overcome such physical linitations. And every year, engineers and physicists patiently establish why the devices do not have the promised properties.

So too in Computing. Over the last several decades, many schemes have been proposed for overcoming the formal limitations on Computing deriving from undecidability results and the Church-Turing thesis. Such schemes typically challenge either the Halting problem, the applicability of Church-Turing to new formalisms or the limitations to computation deriving from the physical realisations of computers.

In this paper, we briefly survey such challenges to the Church-Turing orthodoxy, and then discuss in more detail how Physics actually limits the computing devices we can construct, and the promise of novel yet realisable physical computing configurations.

2. RUMOURS OF HYPERCOMPUTATION

There have been many attempts to articulate new systems for computability that are claimed to transcend the limitations of Church-Turing systems, termed for example *hypercomputing* or *super-Turing*. Copeland (2002) provides a thorough summary. The alleged hypercomputing systems are based, we contend, on either a confusion of calculi with computing machines or on unrealisable physics.

Proponents of hypercomputation require the existence of actual as opposed to potential infinities. Aristotle (1983) made a distinction between *potential* and *actualised* infinities. A potential infinity is one that corresponds to a process that could go on for ever but which can necessarily never be completed. In contrast, an actualised infinity must be concretely realised at some time and place - an inherently contradictory notion. Some hypercomputing schemes involve an actually infinite physical computing resource (Wegner & Eberbach (2004); Beggs & Tucker (2006)) others infinite amounts of time in which to perform computations (Etesi & Németi (2002); Hamkins (2002)). They may also require instantaneous communication between computing devices or the components of these devices (Copeland (2002)), or they invoke infinitely small periods of time for information to pass between separatated communicators. Or they may involve several such infinities at once. For example the proposal by Wegner & Eberbach (2004) that the πcalculus is hypercomputational requires both an infinite number of processes, and that information can be transmitted between processes in zero time (Cockshott & Michaelson (2007)).

The models of physics used by many of the hypercomputing proposals(Beggs & Tucker (2007, 2006); Bournez & Cosnard (1995)) are classical rather than quantum. Smith (2006a) has shown that in a world of point masses and Newtonian physics, the Church Turing thesis would be invalid, but with more realistic laws of motion, it holds. The proposal to use time dilation effects of rotating black holes to allow infinite computations (Németi & David (2006); Etesi & Németi (2002)) is at least relativistic but, among many other difficulties (Cockshott et al. (2008b)), ignores the finite lifetime of black holes according to quantum theory. Kieu (2003) proposed a quantum process whereby a Turing in-computable problem could be solved by adiabatic relaxation to a ground state. This has been subsequently criticised by Smith (2006b) and Hagar & Korolev (2006) as involving arbitrarily small energy measurements or uncomputably long relaxation times.

3. PHYSICS SUGGESTS PHYSICAL LIMITS TO COMPUTING

Any real-world computing device is governed, and therefore limited, by the laws of physics. Electronic digital systems have evolved rapidly in the last half century and will undoubtedly continue to do so for some time to come, but beyond the near term, we can only guess at the future path of computing technologies, their potential capabilities and how these will be achieved. Although the machines of the distant future may harness physics that has yet to be discovered, such as the anticipated theory of quantum gravity, if we accept that the currently known laws of nature are universal, we can still place bounds on what these systems will be able to do.

In what follows it must be borne in mind that any computer is a physical system; conversely, any physical system governed by mathematical laws that determine its evolution in time, is potentially a computer, albeit one that may be of no practical use in solving problems of

[1]Though delimiting the carbon footprints of computers and computations may well soon become a significant practical concern

concern to humans. The bounds that we can place on such systems are completely general but, in reality, may only be closely approached in what would currently be seen as highly “exotic” scenarios. For example Lloyd & Ng (2007)suggests the use of black holes as extreme quantum computing devices, even though no theory governing the temporal evolution of such objects is currently known and the hypothetical technology that might be required to “program” them and read the results has not been and, indeed, may never be, invented

The final limits to what can ever be computed physically, are constrained by the laws of nature and, ultimately, by the total amount of resource (including time) available in the physical universe. Here we will look at two crucial issues: the maximum rate at which elementary operations can ever be performed and the maximum amount of information that can ever be stored. Quantum mechanics limits the speed at which elementary operations can be performed and, as shown by Margolus & Levitin (1998)this limit is related to the total energy available to the computational system. In this context, an elementary operation is one which moves a system from one quantum state to an orthogonal one (for example flipping a qubit). The so-called Margolus-Levetin Theorem is derived by appealing to the energy-time uncertainty relation and shows that a system with total available energy, *E*, can perform at most $\frac{2E}{\pi\hbar}$ elementary operations/second. If the system performs multiple operations concurrently, the available energy is simply subdivided and each operation accordingly takes longer. Parallelism ultimately gains nothing in terms of speed at this limit and the degree to which it makes sense to use concurrency will depend on the actual physical extent of the system: the more extensive the system the greater the parallelism required to exploit its resources, due to the communication constraints imposed by the finite speed of light .

At the same time, theoretical physics places bounds with varying degrees of strictness on the maximum information content that can be stored in a region of space. Some of these bounds, such as that established by Gour (2001), are restricted to systems of certain types (in this case thermodynamically extensive ones) but are relatively exacting, while others are more general, but laxer. One of the most interesting such limits is the *holographic bound*, first deduced by Susskind (1995), which shows that no object can have higher entropy than the black hole that would result if the object were to undergo collapse. Since the information entropy, *S*, of a black hole is given by,

$$S = \frac{A}{4l_p^2 \ln 2}\text{bits}$$

, where *A* is the surface area of its event horizon and l_p is the Planck length (1.6 x 10^{-35} m), this is also an upper limit for the entropy of the original object. The curiosity here is that the absolute limit on information storage appears to be proportional not to the volume of space which the object takes up, but rather the surface area enclosing that volume.

The holographic bound applies (with equality) to black holes but is not attainable by ordinary objects. However, in the case of such objects, we can, in fact, find a more demanding limit by applying a principle known as the generalised second law of thermodynamics (“generalised” in the sense that it includes the concept of black hole entropy). This latter limit was first established by Beckenstein (1981)and is known as the *universal entropy* or *Beckenstein bound*. It states that any system of rest energy, *E*, contained within a volume of space of radius *R* has an information capacity no greater than

$$\frac{2\pi RE}{\hbar c \ln 2}\text{bits}$$

. The Beckenstein and holographic bounds coincide for a black hole but the former is lower, and therefore more informative, for ordinary systems. A system of volume 1 litre and a mass of 1kg, for example, according to the Beckenstein bound, could store no more than 10^{43} bits. Although this is significantly lower than the system’s holographic bound (˜10^{70} bits), Gour’s limit

$$\left(\frac{ER}{\hbar c}\right)^{\frac{3}{4}}$$

is more exacting still and shows that with these parameters the capacity is no more than about 10^{32} bits (almost 40 orders of magnitude below the holographic limit) Gour (2003).

Again it should be emphasised that these limits on processing speed and storage are absolute maxima and apply regardless of technology or physical scenario. Lloyd (2000, 2002)has speculated about what types of macroscopic systems might actually attain these limits. In Lloyd (2000)he discusses how a 1 kg system might attain the Margolus-Levetin limit dictated by a near complete conversion of its rest mass into energy: he calls this device, capable of up to 10^{51} ops/sec the “ultimate laptop” although its nature (running at a temperature of about 10^9K) would make it a rather uncomfortable travelling companion! Its speed is not affected by the volume it occupies but the storage capacity and degree of parallelism is. The bit flip time (operations per bit per second) is reduced to its ultimate limit by collapsing the 1kg mass inside its Schwarzschild radius. This causes the storage capacity to reduce to the holographic bound of a 1kg black hole but reduces the time for a signal to cross the system and thus its level of parallelism.

Lloyd (2002)carries the reasoning further to estimate the total computational capacity of the entire universe. On the basis of some, admittedly uncertain, assumptions about the resources available, he estimates the Universe to have entropy of about 10^{90} bits. This rises to about 10^{120} if the *dark energy* postulated by current cosmological theories is taken into account but, as pointed out inLloyd & Ng (2007), these extra bits can have taken no real part in the universal computing process as their bit flip times are too long. Applying Margolus-Levitin, he calculates that the universe is capable of the equivalent of no more than 10^{106} single bit operations per second and that it has performed the equivalent of 10^{123} such operations since its creation, assumed to be about 10^{10} years ago.

An interesting question relates to why these processing and storage limits manifest themselves at all and what they imply for computation. The answer seems to lie with the relationship between the observer and the observed world, a relationship mediated by the fundamental impossibility of measurement of infinite precision. We can conceptualise systems capable of computing with real number quantities (for example a quantum computer is modelled with qubits which are parameterised by unconstrained real values and evolve continuously in time) but, even if physical systems do in fact function “objectively” in this way (this is a metaphysical question given that we are, by definition, incapable of escaping from our status as observers), we can only exchange information with them according to the principles of quantum mechanical measurement theory and these principles lead inexorably to the constraints outlined above. It should be said here that, carrying this reasoning further, it is possible to question whether it is valid to apply the Margolis-Levetin and holographic bounds, both explicitly derived from a fundamentally epistemological theory like quantum mechanics, to a system like the universe which (it is assumed) contains any observer. However, the conclusion is undoubtedly interesting and, unimaginably huge though the numbers arrived at may be, the fact remains that they are undeniably finite. In short, material systems, as far as we can ever know them, are subject to finite computational limits and while these limits may seem far off when compared to today’s technologies, they place an ultimate ceiling on what can ever be achieved using computers.

4. WHAT IS FEASIBLE

A reader of the previous sections could be forgiven for thinking we were hostile to the idea of non-classical computation. This is not the case. We think that once the mirage of hyper-computation is ignored, there are several real paths that the computer science community could profitably explore. We will look at some of these below, but first, it is worth examining some of the economic constraints that impinge on the development of new computing techiques.

Computing took off as an industry in the 1950s because Turing and von Neumann had shown that you could build one general purpose machine which could then be applied to a large class of problems. This universality justified the very high capital cost of developing the machine. In the computer industry, we see a very clear example of Babbages old principle that copying technologies lie at the heart of mass production (Babbage (1832)). Once a computer has been designed, and once these designs have been materialised in photo-lithography masks, subsequent copies can be turned out cheaply. The design is very expensive but the marginal copies are cheap. The same principle is evident here as was exhibited with the invention of the printing press. The printing press is a universal machine for producing any book, the UV litho plant is a universal system for producing any computer chip. The plant is increasingly expensive; a new factory at 90nm technology on 300mm wafers, has a capital cost of $2-3Billion (Polcari (2005)). It is thus essential that the market for the product be as large as possible. The von Neumann design achieves a large market by virtue of its universality. If new, non-classical computing is to achieve a significant impact it must either:

1. Offer a superior form of universal computing, in terms of speed or power consumption.
2. Address some different mass market for which economies of scale are possible.
3. Address some niche application for which large sums of money are available.

We will examine technical options in the light of the above principles. We argue that the greatest gain from non-classical computation will occur in applications that are related to simulating physical systems on some sort of grid. These are inherently highly parallel, and on a conventional machine the simulation is done by mapping the spatial grid of the physical system onto the one dimensional structure of machine addresses, using various forms of multi-dimensional arrays. The limited bandwidth to memory inevitably slows this down.

Higher performance can be obtained by machines that are specifically designed to act as simulators and which map each locality in the physical system being studied onto a spatial locality on the hardware. The dynamical laws governing the evolution of the physical system are then modeled by tailored dynamical laws which the hardware evolves under.

We can envisage a classification of such systems as follows

	Analogue	Digital
Classical	general purpose analogue simulators	cellular automata machines
Quantum	analogue quantum simulators	digital quantum simulators

Classical systems are those in which the state variables take on definite values. Quantum systems are those where the state variables can exist in superposed states. In analogue systems the state variables take on approximate values from a continuous range, in Digital systems the state variables take on exact values from a discrete range.

Below we will look at some possible technologies.

4.1. Special purpose silicon hardware

Existing CMOS processes are the result of a long and costly technical evolution. They thus offer the most likely starting point for new computing paradigms, since a paradigm that uses silicon technology can be manufactured on existing plant and equipment.

4.1.1. Computing with FPGAs and how this differs from von Neumann computing

Field Programmable Logic Arrays have been a mass market application of silicon technology for over 20 years. They address universality in several ways. For a start, they can be fabricated using the same processes as are needed for standard components like CPUs and DRAMs. Next, the individual chip as it comes out of the factory can be applied to a multiplicity of potential uses. It can be configured to implement any logic circuit up to some bound set by the number of gates on the chip. Usually this universality is applied to some specific use or algorithm for which the individual chip will subsequently be dedicated (Gokhale & Graham (2005)). But it is quite feasible to configure part of the array as a conventional instructionset processor like the NIOS from Altera (Plavec et al. (2005)), or the PicoBlaze from Xilinix. But such ’soft’ cores typically are slower and use more power than conventional ones (Lysecky & Vahid (2005)). This is because the indirect implementation of logic gates via lookup tables, as is done on FPGAs is less efficient than their direct implementation in transitors. Thus to gain an advantage with FPGAs one must find a problem domain where direct implementation of an algorithm in hardware still pays off despite the fact that you will be using slower basic gates than on a conventional processor.

For integrated circuit processors, the von Neumann bottleneck is an abstract reflection of the fact that the processor must communicate with main memory via a limited number of pins on the data bus. Because they face similar pin constraints, FPGAs suffer from their own version of the von Neumann bottleneck. They work best for algorithms that have a naturally limited input output rate. They are particularly suited to stream processing. Video stream processing for example, where the data rate is set by the camera. Another example is those SAT solvers where the entire computational search is done using onboard resources (Cockshott et al. (2008a)).

4.1.2. Cellular automata machines

An earlier generation of FPGAs allowed a different form universal computation : two dimensional cellular automata (Gray & Kean (1989)). The latter are well known to be universal computers (Wolfram (1984, 2002)). Cellular automata can be used to model any physical process with local interaction defined on a manifold with dimensionality equivalent to that of the cellular automaton. For silicon based systems we are effectively limited to 2D automata, and simulating 2D processes (Dunn & Milne (2004); Fu & Milne (2003); Milne et al. (1993)). When an FPGA or array of FPGAs is configured as a general cellular automaton, very high computational rates can be achieved (Shaw & Milne (1993)) relative those achievable on a von Neumann computer of the same physical size using contemporaneous chip technology. The key features of this approach are to map each region of the manifold under simulation to a locality of the silicon, and to store all state associated with that region in local registers. The state transition rules can then be implemented in fast combinational logic. Since storage is in local registers, state updates for all regions of the manifold can be simultaneous.

This approach was once synchronously scalable by adding additional FPGAs to the array, but in recent years the grid of automata that can be implemented on a chip has outgrown the available pinout, necessitating the use of multiplexing. This makes the approach slightly less promising. It is also notable that when working with current FPGAs the tool chain, starting as it does with an abstract language VHDL, does not give the ready control over layout geometry that one really wants for cellular automata work. Because the layout is done automatically, and because automated layout is potentially NP hard, the layout phase of compiling the design to the chip can be rather slow.

4.1.3. Possible analogue FPGAs

> During the 1950s and 1960s electronic and hybrid analogue computers were at the heart of modelling such technological systems as aerospace and industrial plant control. Heated debates took place about the relative virtues of analogue and digital computers. (Bissell (2004))

For certain application areas economic modelling for example (Velupillai (2003)), even hydraulic analogue machines proved useful right down till the early 1970s. Analogue computers had, for a considerable period, a marked advantage in terms of speed.

> At Project Cyclone digital computers were [...] used to verify the accuracy of results from the analog computer. [...] In the simulation of a guided missile in three dimensions, the

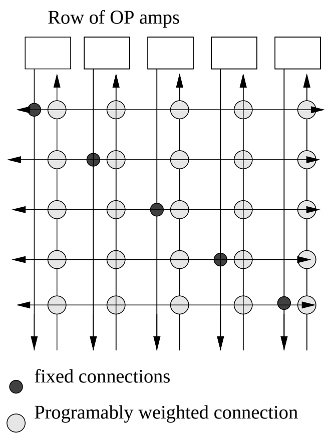

Figure 1: *A fully interconnected array.*

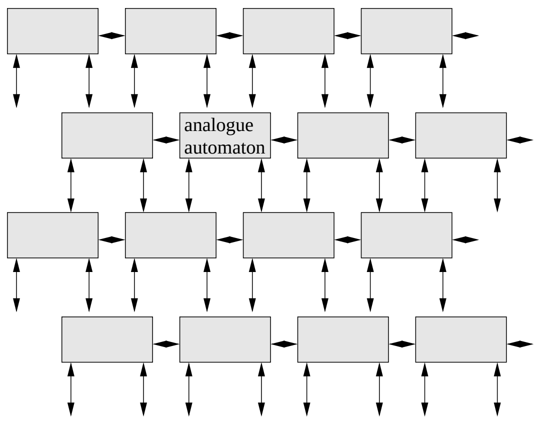

Figure 2: *Analogue automata on a hexagonal lattice on a chip*

> average runtime on the analog computer facility was approximately one minute. The check solution by numerical methods [...] took from 60 to 130 hours to solve the same problem. (Small (1993), p. 11)

The accuracy attainable by an analogue machine is inherently limited, but in many classes of applications this was not a serious issue, since the raw observations on which the computation was based had significant uncertainty.

The computers were programmably by rewiring op amps and other components using plug boards. In many ways this is analogous to the rewiring that one now does on FPGAs. The latter are universal digital circuits. One could in principle design classes of universal analogue circuits that would span the equivalent analogue computing space. One could envisage an architecture like that shown in Figure 1 being used to implemement a general analogue computer on a chip. Because a fully general machine requires a potentially full interconnect network, the area of such a chip is dominated by the programmable interconnect and grows as n^2 in the number of variables handled n.

At the other extreme one can envisage a network of analogue components in a square grid, each of which communicates a few analogue state variables with its 4 or 6 nearest neighbours (square or hexagonal tessellation). Internally the cell might have two or three analogue state variables which could be controlled by some programmable differential equation driven by the states of its neighbours. This configuration, would be a universal analogue cellular automaton.

These sorts of architectures could certainly be built, and would to a large extent capitalise on the existing infrastructure for the construction of digital circuits. Because the latest processes are always reserved for the newest designs of the main semi-conductor companies, any such analogue machine would be made using older processes. Like any experimental chip it would have larger feature sizes and lower integration than the most advanced digital products. The questions then are:

1. Whether a convincing case could be made that there exist applications areas where analogue computers are again applicable. One could imagine applications for configurations like Figure 2 in areas like spin-glass models, fluid dynamics, diffusion modelling and the like.
2. Whether programming tools could be developed that would enable very large analogue circuits to be effectively used. In this context Bissell (2004) points out that modern interactive tools like Matlabs Simulink present a model very close to that of analogue computers. One would have to combine this with some sort of tesselation and tiling tool for models like that in Figure 2.

4.2. Quantum computing

The idea of quantum computing originates with the observation by Feynmann (1999) that quantum mechanics was inherently hard to simulate on digital computers because the digital representation of a quantum system grows as the tensor product of the basis states of its component parts. Thus for n subsystems each with two basis states, we have a state space of 2^n. The evolution of the system, if represented by a matrix operator, then requires a storage space of 2^{2n} and each evolution step is also of order 2^{2n}. He proposed therefore, that it would be better to try to simulate a given quantum system **A** with another quantum system **B**. Provided that a one to one mapping can be established between the states and dynamics of the the two quantum systems, the superposition of states in **B** can then directly simulate the superposition of states in **A** without the exponential explosion that we experience when applying classical techniques of computation. From this suggestion he went on to derive a set of reversible logic gates that could in principle operate according to quantum mechanics. The potential computational universality of such machines was soon established (Deutsch (1985)).

4.2.1. Quantum digital computers
In over two decades since Deutsch's key paper, a significant effort has gone into investigating both potential physical implementations of quantum computers and algorithms that might run on them. Most practical effort has concentrated on the *quantum circuit model* which is computationally equivalent to Deutsch's *Quantum Turing Machine* (Deutsch (1985)). More recent alternative models such as the *quantum adiabatic computer* have also been shown to be computationally equivalent to the circuit model (Aharonov et al. (2008)) to within polynomial overhead.

The circuit model relies on the application of a sequence of unitary transformations, called *quantum gates*, to a register of *n qubits*, the quantum generalisation of the one bit store, capable of being in general superpositions (parameterised by real values) of the two basis states, 0 and 1. The sequence of transformations can be conceptualised as being ordered by 'quantum wires', although these are not real wires and, in fact, correspond physically, to particle transfers or just unitary time evolution (as determined by the time-dependent Schrodinger equation). Multiple input gates cause the qubits to become entangled and eventually the quantum circuit (gates plus wires) produces an output which can then be subjected to a quantum measurement relative to some chosen basis. The idea is to design the circuit so as to implement the logic of some desired computation. It is easy to show that any classical logic operation can be implemented in this way; however, although this implies that any classical algorithm can be simulated on a quantum computer, in general no speed-up will be observed. It is also possible to demonstrate that any classical probabilistic algorithm can be simulated on a quantum computer.

The power of quantum computation comes from the parallelism inherent in entanglement. In general the output is a superposition, $|\psi\rangle$ of basis states, $|\phi_i\rangle$ from the selected measurement basis (often the tensor product of the single qubit basis); however, a quantum measurement, when conducted, will obtain only one result, at random, with a probability equal to the square of the amplitude of $|\phi_i\rangle$ in $|\psi\rangle$ (namely $\langle\phi_i|\psi\rangle$). The key to a successful quantum algorithm is to ensure that the result of the computation is encoded in these amplitudes. It follows, however, that a quantum computer will yield a correct answer only with a certain probability and that to retrieve it will, in general, require multiple runs.

As might be suspected from this, successful quantum algorithms are hard to design. To be useful, obviously, a quantum algorithm must offer some significant advantage over an equivalent on a classical machine and to engineer this is not a trivial exercise. The most celebrated example is undoubtedly Shors quantum factoring algorithm (Shor (1999)) which can find the prime factors of an integer in polynomial time, exponentially faster than any known classical technique. Other important quantum algorithms, like Grovers search algorithm (Grover (1997)), are faster than any classical equivalent, although not exponentially so, but notable examples like these are still relatively few. In fact Shor himself has expressed disappointment at the limited success so far and examines possible reasons in Shor (2004)

In terms of computational complexity, factoring is in complexity class NP but it is not NP complete and, despite much effort, nobody has be en able to demonstrate that quantum computers can solve problems in the latter category in polynomial time. Shor (2004) is pessimistic, pointing out that there is a proof that a quantum computer cannot search a space of size N in less than O($\sqrt{N}$) time. As for hypercomputation, attempts have been made to use quantum computing (in adiabatic guise) to solve Hilbert's Tenth Problem (Kieu 2004), a famous challenge to construct a general method for deciding whether a given Diophantine equation has an integer solution. It is known that this problem can be solved if and only if the Halting problem can be solved, so success would herald the potential creation of a hypercomputer. However Kieu's solution, which harnesses the quantum adiabatic theorem, has been convincingly challenged by various authors (see e.g. Hagar & Korolev (2006)).

Quantum digital computers can potentially significantly outperform classical equivalents in some problems. However this superiority depends on the controllable entanglement of many qubits and, as is well-known, quantum theory shows that such control can only be achieved if the entire system can be prevented from experiencing decoherence. Yet all quantum systems are subject to interaction with the environment and it is a significant challenge to minimise such interaction for long enough to allow a computation to complete. Inevitably some decoherence will occur and this has the effect of introducing noise and thus error into the process. Bounding this error is one of the key issues that must be confronted if quantum computing is to become a practical proposition. Several technologies are being extensively researched as potential candidates for such practical implementation and it is not yet clear which will prove the most successful. DiVincenzo (2000) lists the requirements any such technology must meet: a scalable physical system with well characterised qubits; the ability to initialise qubits; long decoherence times; a universal set of implementable quantum gates; and a qubit-specific measuring capability.

Other (computationally equivalent) approaches to quantum computing have been suggested which attempt to circumvent the problem of decoherence. The adiabatic approach (Farhi et al. (2000)), based on controlled adiabatic Hamiltonian evolution, has already been mentioned, and has been the subject of some effort. It is the basis not only for Kieu's hypercomputation claim but also for the technology behind the controversial announcements of quantum computer implementations by the Canadian company *D-wave*. More speculative still is *topological quantum computing* (Freedman et al. (2003)) which is based on the use of *anyons* (Wilczek 1982), 2-dimensional quasi-particles channelled through gates which are formed from *braids* of world-lines in a 3-dimensional space-time. Quasi-particles are pseudo-physical mathematical constructs of which the commonest examples are the holes of semiconductor physics and the phonons of solid-state physics; braids are a topological generalisation of the familiar twisted-string forms of the same name. Theoretically a topological quantum computer would be much more resistant to decoherence than one based on quantum circuits. However, at present this line of investigation remains at an even more hypothetical level of development than the rest of what is still a discipline some way from bearing practical fruit.

4.2.2. Quantum simulators

As we said above, the simulation of quantum systems was the first use proposed for quantum computers. Lloyd (1996) demonstrated that a digital quantum computer can in principle simulate any quantum system that evolves by means of local interactions. More significantly he showed that if we wish to simulate a quantum system operating for time t the simulation time on a quantum computer will be linear in t.

A quantum computer can thus act as a universal quantum simulator. When operated in this mode, the requirements for error resilience and resistence to decoherence are less than for numerical applications of quantum computers since the decoherence in the simulator can be used to model real decoherence in the system under study. This potentially allows techniques with a shorter decoherence time to be used than would otherwise be required. Llloyd showed that simulation of a quantum system with *N* variables that evolves according to a local Hamiltonian requires a number of steps that is linear in *N*. This is obviously a huge improvement on the classical process which involves the exponentiation of a matrix of size 2^{2N}.

As we said above the process of actually developing a viable quantum computer technology has been slow. General purpose quantum computers are reminiscent of thermonuclear power: feasible in principle, always promised some time in the next 20 years. But as Buluta & Nori (2009) report, the less exacting requirements of simulation are more permisssive. They report arrays of quantum dots and superconducting circuits as being among the technologies under active development. Perhaps the most promising lines of development are quantum devices using superconducting circuits. In these large numbers of pairs of electrons can condense into a single state. Because the superconducting state exhibits macroscopic degrees of freedom it exhibits better coherence than qubit systems using individual atoms which have only microscopic degrees of freedom.

It is possible to construct artificial 'atoms' with precisely tuned characteristics (You & Nori (2006)). The advantage of such solid state systems from a production standpoint is that they can take advantage of a well established engineering capability for microfabrication developed for the semi-conductor industry. It seems likely at present that such supercomputing quantum simulators will be the first devices to be put to practical use.

5. CONCLUSIONS

Another thing they couldn't stand was the perpetual failure they encountered in trying to construct a machine which could generate the infinite improbability field needed to flip a spaceship across the mind-paralysing distances between the furthest stars, and in the end they grumpily announced that such a machine was virtually impossible.Adams & Stamp (1997)

We have surveyed fundamental limits to information processing in physics and, by implication, to computational machinery. Of course, a paradigm shift as profound as relativity theory and quantum mechanics might well make these limits seem as mundane as those of Newtonian mechanics and 19th century engineering. Speculation as to where such a shift might come from is a fundamental scientific perogative. But right now we are where we are.

Nonetheless, even in the absence of some new Physics offering warp drives and infinite improbability devices, we should not be discouraged by limitations dictated by our prevailing paradigms, either theoretically or practically. While the elaboration of new models of computatability appears to confirm the Church-Turing thesis, they offer new understandings of how we may more efficaciously express, manipulate and realise computations. And while new physical models of computation are ultimately constrained by established limits to the characteristics of physical systems, and the accuracy to which those characteristics can be measured, we are still a very long way from being restricted by them in any practical sense. We can see already that special silicon purpose hardware and quantum devices, while still very much in their infancy, offer profound changes in how we currently conceive computing, and in the scale and accuracy to which we can model and control the world. And, in the longer term, new substances and devices, even if based solely on greater understanding of current physics and engineering principles, will undoubtedly prove as revolutionary as the now common place chip. We should view the limits to computation and physics, not as sources of frustration or despair, but as challenges to our ingenuity in how to best exploit the vast space of possibilities they bound.

6. REFERENCES

D. Adams & R. Stamp (1997). *The hitchhiker's guide to the galaxy*. Del Rey.

D. Aharonov, et al. (2008). 'Adiabatic quantum computation is equivalent to standard quantum computation'. *SIAM Journal on Computing* **37**(1):166–194.

Aristotle (1983). *Aristotle's Physics*. Clarendon, Oxford.

C. Babbage (1832). *The Economy of Machinery and Manufactures*. London.

J. Beckenstein (1981). 'Universal upper bound on the entropy-to-energy ratio for bounded systems'. *Phys. Rev. D.* **23**:287 – 298.

E. Beggs & J. Tucker (2006). 'Embedding infinitely parallel computation in Newtonian kinematics'. *Applied mathematics and computation* **178**(1):25–43.

E. Beggs & J. Tucker (2007). 'Can Newtonian systems bounded in space, time, mass and energy compute all functions?'. *Theoretical Computer Science* **371**(1):4–19.

C. Bissell (2004). 'A great disappearing act: the electronic analogue computer'. In *IEEE Conference on the History of Electronics Conference Proceedings*, pp. 28–30.

O. Bournez & M. Cosnard (1995). 'On the computational power and super-Turing capabilities of dynamical systems'. Tech. Rep. 95-30, Ecole Normal Superior de Lyons.

I. Buluta & F. Nori (2009). 'Quantum Simulators'. *Science* **326**(5949):108.

P. Cockshott, et al. (2008a). 'A Hardware Relaxation Paradigm for Solving NP-Hard Problems'. In *International Academic Conference Visions of Computer Science*. BCS.

P. Cockshott, et al. (2008b). 'Physical constraints on hypercomputation'. *Theoretical Computer Science* **394**(3):159–174.

P. Cockshott & G. Michaelson (2007). 'Are There New Models of Computation? Reply to Wegner and Eberbach'. *The Computer Journal* **50**(2):232.

J. Copeland (2002). 'Accelerated Turing Machines'. *Minds and Machines* **12**:281–301.

D. Deutsch (1985). 'Quantum theory, the Church-Turing principle and the universal quantum computer'. *Proceedings of the Royal Society of London A* p. 97..117.

D. DiVincenzo (2000). 'The Physical Implementation of Quantum Computation'. *Arxiv preprint quant-ph/0002077*.

A. Dunn & G. Milne (2004). 'Modelling wildfire dynamics via interacting automata'. *Lecture notes in computer science* **3305**:395–404.

G. Etesi & I. Németi (2002). 'Non-Turing Computations Via Malament–Hogarth Space-Times'. *International Journal of Theoretical Physics* **41**(2):341–370.

E. Farhi, et al. (2000). 'Quantum computation by adiabatic evolution'. *Arxiv preprint quant-ph/0001106*.

R. P. Feynmann (1999). 'Simulating physics with computers'. In A. Hey (ed.), *Feynmann and computation: exploring the limits of computers*, pp. 133–153. Perseus Books, Cambridge, MA, USA.

M. Freedman, et al. (2003). 'Topological quantum computation'. *Bulletin of American Mathematical Society* **40**(1):31–38.

S. Fu & G. Milne (2003). 'Epidemic modelling using cellular automata'. In *Proc. of the Australian Conference on Artificial Life*.

M. Gokhale & P. Graham (2005). *Reconfigurable computing: accelerating computation with field-programmable gate arrays*. Springer Verlag.

G. Gour (2003). 'Extensive entropy bounds'. *Physical Review D* **67**(12):127501.

J. Gray & T. Kean (1989). 'Configurable hardware: A new paradigm for computation'. In *Decennial CalTech Conference on VLSI*, pp. 277–293.

L. Grover (1997). 'Quantum mechanics helps in searching for a needle in a haystack'. *Physical Review Letters* **79**(2):325–328.

A. Hagar & A. Korolev (2006). 'Quantum hypercomputability?'. *Minds and Machines* **16**(1):87–93.

J. Hamkins (2002). 'Infinite Time Turing Machines'. *Minds and Machines* **12**(4):521–539.

T. D. Kieu (2003). 'Quantum Algorithm for Hilbert's Tenth Problem'. *International Journal of Theoretical Physics* **42**:1461 – 1478.

S. Lloyd (1996). 'Universal quantum simulators'. *Science* **273**(5278):1073–1078.

S. Lloyd (2002). 'Computational capacity of the universe'. *Physical Review Letters* **88**(23):237901.

S. Lloyd & Y. Ng (2007). 'Black hole computers'. *Special Editions* **17**(1):82–92.

ï. Lloyd (2000). 'Ultimate physical limits to computation'. *Nature* **406**:1047–1054.

R. Lysecky & F. Vahid (2005). 'A study of the speedups and competitiveness of FPGA soft processor cores using dynamic hardware/software partitioning'. In *Proceedings of the conference on Design, Automation and Test in Europe-Volume 1*, pp. 18–23. IEEE Computer Society Washington, DC, USA.

N. Margolus & L. Levitin (1998). 'The maximum speed of dynamical evolution'. *Physica D: Nonlinear Phenomena* **120**(1-2):188–195.

G. Milne, et al. (1993). ‘Realising massively concurrent systems on the SPACE machine’. In *IEEE Workshop on FPGAs for Custom Computing Machines, 1993. Proceedings*, pp. 26–32.

I. Németi & G. David (2006). ‘Relativistic computers and the Turing barrier’. *Applied Mathematics and Computation* **178**(1):118–142.

F. Plavec, et al. (2005). ‘Experiences with soft-core processor design’. In *Proc. of the 19th IEEE Conference on International Parallel and Distributed Processing Symposium*.

M. Polcari (2005). ‘Collaboration: The Semiconductor Industry’s Path to Survival and Growth’. In *AIP Conference Proceedings*, vol. 788, p. 3. IOP INSTITUTE OF PHYSICS PUBLISHING LTD.

P. Shaw & G. Milne (1993). ‘A highly parallel FPGA-based machine and its formal verification’. *LECTURE NOTES IN COMPUTER SCIENCE* pp. 162–162.

P. Shor (1999). ‘Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer’. *SIAM Review* **41**(2):303–332.

P. Shor (2004). ‘Progress in quantum algorithms’. *Quantum Information Processing* **3**(1):5–13.

J. Small (1993). ‘General-purpose electronic analog computing: 1945-1965’. *IEEE Annals of the History of Computing* **15**(2):8–18.

W. Smith (2006a). ‘Church’s thesis meets the N-body problem’. *Applied mathematics and computation* **178**(1):154–183.

W. D. Smith (2006b). ‘Three counterexamples refuting Kieu’s plan for “quantum adiabatic hypercomputation” and some uncomputable quantum mechanical tasks’. *J.Applied Mathematics and Computation* **187**(1):184–193.

L. Susskind (1995). ‘The world as a hologram’. *Journal of Mathematical Physics* **36**:6377.

K. Velupillai (2003). ‘Essays on Computable Economics, Methodology and the Philosophy of Science’. Tech. rep., Universita’ Degli Studi di Trento - Dipartimento Di Economia.

P. Wegner & E. Eberbach (2004). ‘New Models of Computation’. *Computer Journal* **47**:4–9.

S. Wolfram (1984). ‘Computation theory of cellular automata’. *Communications in Mathematical Physics* **96**(1):15–57.

S. Wolfram (2002). *A New Kind of Science*. Wolfram Media, Inc., Illinois.

J. You & F. Nori (2006). ‘Superconducting circuits and quantum information’. *Arxiv preprint quant-ph/0601121* .