Five views of hypercomputation

BRUNO LOFF\textsuperscript{1,2}, JOSÉ FÉLIX COSTA\textsuperscript{1,2†}

\textsuperscript{1} Instituto Superior Técnico, Dept. of Mathematics, Technical University of Lisbon
\textsuperscript{2} CMAF, Complexo Interdisciplinar, Universidade de Lisboa
Lisboa, Portugal

We survey different approaches to the study of hypercomputation and other investigations on the plausibility of the physical Church–Turing thesis. We propose five theses to classify approaches to this area of investigation.

Halloween.

“You don’t know, do you?” asks Carapace Clavicle Moundscrounching out of the pile of leaves under the Halloween Tree. “You don’t REALLY know!”

— Ray Bradbury, The Halloween Tree

\textsuperscript{*} email: bruno.loff@gmail.com
\textsuperscript{†} email: fgc@math.ist.utl.pt
1 INTRODUCTION

In (7; 8) Martin Davis criticises the relevance of so-called hypercomputation. Is hypercomputation a new theory to help us understand the mathematics of hyperdegrees, or is it a theory about concrete computation in the physical world? We review the concept of hypercomputation to realise that five main views, or trends, can be found in the literature.

People outside the field mainly think that hypercomputation is based on certain properties of real numbers, that models are endowed with hypercomputation by the fact that they incorporate real parameters, using the infinite amount of information contained in some incompressible real number.

This view is wrong! The hypercomputational contents of some computational models have nothing to do with the infinite amount of information that may be contained in real numbers. Davis’ paper (7) has been misused by others, both in oral presentation and paper citation. The relevance of the real numbers is low, and the hypercomputational contents of these models arise from treating the reals as oracles, or advice: it is only possible to use a finite part of the oracle (or real number) during a finite computation.

In these pages we survey some models of hypercomputation and attempt to clarify some misunderstandings in research. We will organise our discussion into five sections, each of which describes one main trend in the field. For each of these trends, some statement is made about the existence of hypercomputation. In Section 2 we overview hypercomputation as the study of non-simulable physical phenomena. In Section 3 hypercomputation is performed by abstract physical devices that have a super-Turing power but which can not be finitely specified. There are proposed hypercomputational physical systems which can be finitely specified, and these are described in Section 4. In Section 5, hypercomputation refers to the study of non-algorithmic sources of information which may exist in nature. Finally, in Section 6, we report that hypercomputation is sometimes considered an essential part of the structure of the Universe.

The reader will notice that we do not argument in favour or against any form of hypercomputation, nor do we discuss our own opinion on this matter. There are other places and other articles where it is more appropriate to do so. The purpose of this article is to provide an introductory overview of hypercomputation, to clear up some common misinterpretations, and to offer a cartography of the subject.
2 HYPERCOMPUTATION AS UNSIMULABLE PHENOMENA

Our first view of hypercomputation emerges from the detailed study of physical theories with the aim of finding whether computers can or cannot simulate what these theories intend to model. It is based on the following:

Thesis S (for ‘simulation’). There are non-simulable phenomena in physical theories. Thus (maybe) physical reality has hypercomputational content.

The work we will review is grounded on Turing’s definition of computable real numbers and the Lacombe–Grzegorczyk characterisation of computable real functions, and it is worthwhile to recall the following definitions:

Definition 1 (I) A sequence of rational numbers, \((q_n)_{n \in \mathbb{N}}\), is called computable if for some 1-ary total recursive functions \(a, b, c\):

\[
q_n = (-1)^a(n) \frac{b(n)}{c(n)}.
\]

(II) \(r \in \mathbb{R}\) is a computable real number if some computable sequence of rational numbers, \((q_n)_{n \in \mathbb{N}}\), is such that, for all \(n \in \mathbb{N}\), \(|q_n - r| \leq 2^{-n}\).

(III) \(X \in \mathbb{R}^k\) is a computable tuple of real numbers if every element in the tuple is a computable real number.

(IV) \(f : \mathbb{R}^k \to \mathbb{R}\) is a computable function if there are three recursive functionals \(A, B, C : \mathbb{R}^k \times \mathbb{N} \to \mathbb{N}\) such that, for every tuple \(\vec{x} \in \mathbb{R}^k\),

\[
\left| (-1)^{A(\vec{x}; n)} \frac{B(\vec{x}; n)}{C(\vec{x}; n)} - f(\vec{x}) \right| \leq 2^{-n}, \text{ for every } n \in \mathbb{N}.
\]

This is the notion of computable real function that we will use in this section (and only in this section). We can now investigate whether the evolution of a system with computable input and obeying certain physical laws may be uncomputable. In (26) Marian Pour-el and Ian Richards describe a 2-ary computable real function, \(g\), such that none of the (non-unique) solutions of the differential equation

\[
h(x) = 0 \quad \partial_x h(x) = g(x, h(x))
\]

are computable. This result is not tied to any physical theory. In a latter article (27) the same authors describe a certain 3-ary computable function, \(f\), such
that the three-dimensional wave equation

\[
\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2} - \frac{\partial^2 u}{\partial t^2} = 0
\]

with the initial conditions

\[
u(x, y, z, 0) = f(x, y, z) \quad \frac{\partial u}{\partial t}(x, y, z, 0) = 0
\]
gives a unique solution which is not computable. In fact, \(u(0, 0, 0, 1)\) is a non-computable real number. This result is extended in (28) to show that, given a compact set \(D\), \(f\) can be constructed so that \(u\) is not computable in any neighbourhood of any point in \(D\). The uncomputability of the solutions of the wave equation has been thoroughly investigated by Klaus Weihrauch and Ning Zhong in (34). They study the computability of functions in different topological spaces to conclude that if \(f \in C^k(\mathbb{R}^3)^*\) is computable, and all its \(k\)th-order partial derivatives are also computable, then the solution of the wave equation above must be of class \(C^{k-1}(\mathbb{R}^4)\) and all its partial derivatives of order up to \(k - 1\) must also be computable. We can conclude — as Pour-el and Richards already had — that the partial derivatives of \(f\) are not computable, and this is, in fact, where the uncomputability of \(u\) comes from. In regard to the physical feasibility of a wave computer, Weihrauch and Zhong write:

\begin{quote}
In summary, even under very idealising assumptions about measurements and wave propagation in reality, it seems to be very unlikely that the Pour-el and Richards counterexample can be used to build a physical machine with a ‘wave subroutine’ computing a function which is not Turing computable. We may still believe that the [physical] Church–Turing Thesis holds.
\end{quote}

However, Pour-el and Richard’s constructions show that it is not difficult to incorporate non-computable phenomena into physical theory. Roger Penrose (24) also comments on Pour-el and Richards’ results:

\begin{quote}
[...] their ‘peculiar’ kind of initial data is not ‘smoothly varying’ (that is, not twice differentiable), in a way that one would normally require for a physically sensible field.
\end{quote}

Another investigator of Thesis S is Warren D. Smith. In his paper (31) Smith describes a Newtonian system of \(N\) point-masses in 2-dimensional

\* \(C^k(\mathbb{R}^3)\) denotes the space of continuously differentiable functions over \(\mathbb{R}^3\) of degree \(k\).
Euclidean space that cannot be simulated by a Turing machine. We fix the masses of the system to certain rational numbers, allowing such a system to be specified by a tuple of $4N$ real numbers — two real numbers for the position vector and two for the velocity vector, times $N$ particles: we call such a tuple a *specification*. The system follows the law of motion,

$$\ddot{x}_i = G \sum_{j \neq i} \frac{m_j}{\|x_j - x_i\|^3}.$$  

Smith’s results tell us that there is no general algorithm that is, given a computable specification, able to decide the question: *Does any point mass in the system intersect the unit ball centered at the origin in the first second of the system’s evolution?*  

† So being able to specify an $N$-body Newtonian system in 2-dimensional Euclidean space to an arbitrarily high precision is not enough to be able to predict what the system’s behaviour will be. In (31), Smith writes:

*The present paper demonstrates (I claim) that unsimulable physical systems exist in Newton’s laws of gravity and motion for point masses. However, it does not appear to demonstrate, that, if we lived in a universe governed by those laws, we could actually build a device with super-Turing computational power.*

In the next section we will study what such devices may look like, and in section 6 we will give a brief account on the idea that non-computability should be an essential part of any ‘ultimate’ physical theory.

As Edwin Beggs and John Tucker (1) point out, it is important to study the hypercomputational properties of physical theories *even when we know the theory to be false or non-applicable*. A rigorous study of these properties allows us to better understand the sort of phenomena which give rise to non-computability, and thus study other, more strongly believed physical theories with a sharper eye. This should also provide further insight on the applicability of the physical theory itself. Another important purpose of studying these theories is to try and find the necessary and sufficient conditions that make the theory simulable. For instance, Smith, following his exploration of the uncomputability of Newtonian systems, amends the laws of movement to include relativistic phenomena, showing that, under these laws, the $N$-body

† We can assume that if the body does intersect the unit ball in the system then it will also intersect $B(0, 1 - \epsilon)$ for some predetermined $0 < \epsilon < 1$, i.e., Smith’s undecidability result does not stem from the fact that the body may intersect the unit ball arbitrarily near the perimeter.
problem becomes simulable. Beggs and Tucker (2) show that if they demand a specification for the construction of a certain hypercomputational device, then it loses many of its hypercomputational properties. This leads them to consider whether it is justified to add axioms of constructibility to certain physical theories, in order to bound the computational power of devices which are models of the theory.

3 HYPERCOMPUTATION AS COMPUTATION WITHOUT A PROGRAM

The second view we present is based on the idea that hypercomputation can be performed by abstract physical devices, i.e., by machines with an idealised behaviour obeying certain laws of physics. It is nevertheless impossible to program these machines, as their specification cannot be described finitely:

Thesis N (for ‘not programmable’). Hypercomputation is about computations that can not be folded into a program, but can be performed by an abstract physical machine.

The theory of analogue recurrent neural networks (ARNNs) was developed by Hava Siegelmann and Eduardo Sontag (see (30)). Each ARNN consists of a finite number of inputs and units which evolve in discrete time steps. In a network with $N$ units and $M$ inputs the state of each unit is updated at every step, following the rule

$$x_i(0) = 0 \quad x_i(t + 1) = \sigma \left( \sum_{j=1}^{N} a_{ij} x_j(t) + \sum_{k=1}^{M} b_{ik} u_k(t) + c_i \right),$$

where $x_i(t)$ is the state of the $i$th unit at time $t$, $u_k(t)$ is the value of the $k$th binary input at time $t$ and $a_{ij}, b_{ik}, c_i$ are real constants called weights. Above, $\sigma$ is the activation function, given by

$$\sigma(x) = \begin{cases} 
0 & \text{if } x \leq 0, \\
 x & \text{if } 0 < x < 1, \\
1 & \text{if } x \geq 1.
\end{cases}$$

The conventional way of inputting a binary string $w_1 \ldots w_n \in \{0, 1\}^*$ is using two input lines $u_1$ and $u_2$. For every time step $t \leq n$ we set $u_1(t) = w_t$ and $u_2(t) = 1$, and if $t > n$ we set $u_1(t) = u_2(t) = 0$. The output of a binary string is done in a similar way, by means of two chosen units in the network.
ARNNs provide a uniform model for expressing varied computational power. The following table shows the set of languages decidable by an ARNN under different restrictions. It is possible to restrict the set of weights — insisting, for instance, that $a_{ij}, b_{ik}, c_i \in \mathbb{N}$ — or the number of steps the network is allowed to take. Below, $t$ will be a time constructible function.

<table>
<thead>
<tr>
<th>Set of weights</th>
<th>Time restriction</th>
<th>Computational power</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathbb{N}$</td>
<td>none</td>
<td>Regular languages</td>
</tr>
<tr>
<td>$\mathbb{Q}$</td>
<td>none</td>
<td>Recursive languages</td>
</tr>
<tr>
<td>$\mathbb{Q}$</td>
<td>$t$</td>
<td>$\text{DTIME}(t)$</td>
</tr>
<tr>
<td>$\mathbb{R}$</td>
<td>polynomial</td>
<td>$\mathbb{P}/\text{poly}$</td>
</tr>
<tr>
<td>$\mathbb{R}$</td>
<td>none</td>
<td>All languages</td>
</tr>
</tbody>
</table>

FIGURE 1
Computational power of ARNNs under various restrictions.

Up to this point there is no claim for hypercomputation. The original work of Siegelmann and Sontag is, first and foremost, a study in structural complexity, with relevant results for non-uniform complexity classes. There is, we insist, no mention of hypercomputation. Nevertheless, the model has endured severe criticism (7), mostly targeting Siegelmann’s 1995 article *Computation beyond the Turing limit* (29). In this article, and in the expanded version found in (30), Siegelmann shows that ARNNs are computationally equivalent to *analogue shift maps*, a model of computation which extends Cris Moore’s *generalised shift maps* (22). Both of these systems can be implemented by an abstract optical device based on parabolic mirrors, which is set up according to the weights of the network. The device is abstract, because it is conceived using idealised components, such as perfect parabolic mirrors. We discuss the following

**Common misconception.** The device cannot be built because the system must be adjusted with infinite precision (e.g. 10, p. 100).

To understand why this is not the true reason we must explain (roughly) a few details of how we can prove (B). Although not simple to prove, it is straightforward to show schematically that ARNNs with real weights can decide $\mathbb{P}/\text{poly}$ in polynomial time. From (A) above we can see that there must be a neural net which can decide the circuit value problem in polynomial time, call it $\text{NCVP}$. We also know that $\mathbb{P}/\text{poly}$ is the class of languages decidable
by polynomial size circuits. The following proposition, which we can derive from the work by Siegelmann and Sontag, provides the final ingredient:

**Theorem 2** (1) There is a bijective encoding from the set of families of circuits to the 9-Cantor subset of $[0, 1]$:

(2) if $\alpha$ encodes a family $(A_k)_{k \in \mathbb{N}}$ of polynomial size circuits, then the code of the $n$th circuit can be found among the first $p(n)$ digits of the decimal expansion of $\alpha$, where $p$ is a polynomial depending on $(A_k)_{k \in \mathbb{N}}$, and furthermore

(3) we can construct an ARNN — call it $N_\alpha$ — with weights in $\mathbb{Q} \cup \{\alpha\}$ to extract, given input $n$, the code of $A_n$ in a number of steps bounded by $p(n)$.  

![FIGURE 2](image)

**FIGURE 2**

$\mathsf{P/poly}$ with an ARNN.

So given a set in $A \in \mathsf{P/poly}$, we find the real number $\alpha$ that codes for the family of polynomial size circuits which decide $A$, build the net $N_\alpha$ and the net $\text{NCVP}$. Then we build an ARNN which, given input $x$, feeds $|x|$ into $N_\alpha$, and when $N_\alpha$ has extracted $A_{|x|}$, feeds $\langle A_{|x|}, x \rangle$ into $\text{NCVP}$, from where it obtains the output. This is schematised in Fig. 2.‡

The common misconception mentioned above can be dispelled by the following theorem, which tells us that linear precision is sufficient to simulate an ARNN.

**Theorem 3** The output of an ARNN after $t$ steps is only influenced by the first $O(t)$ digits in the decimal expansions of the weights.

This means that, given input $x$, we can simulate a polynomial number $p(|x|)$ of steps of an ARNN in polynomial time, given as advice $O(p(|x|))$ bits

‡ While the figure is not completely faithful to Siegelmann and Sontag’s work — there is no delay unit in the ARNN they present — we decided to favour clarity over accuracy.
specifying the weights of the network. We can simulate ARNNs in P/poly. So in order to make the optical device work correctly on inputs of size $n$ it would only be necessary to adjust the mirrors of the device up to a polynomial precision in $n$.

While this may still be too much precision to allow for implementation, the true reason this device cannot be built in order to do hypercomputation is more fundamental: the digits of the weights themselves cannot be known in advance. Before even considering the technical problem of building the machine with the mirrors precisely adjusted, we stumble upon the evident difficulty of finding out the angle of adjustment itself.

This eliminates the possibility of building such a machine for any useful purpose. Nevertheless we can do the following philosophical thought experiment: imagining that such a machine is build, with the mirrors positioned more-or-less randomly, and then asking if it is performing hypercomputation in some sense.

Another model which follows Thesis N is the scatter machine, recently
introduced by Beggs and Tucker (3). The scatter machine consists of two straight mirrors forming a wedge, a laser and two light detectors laid out as in Fig. 3. The laser can be placed (for instance) in any position of the form $\frac{n}{2m}$. In this case it is possible to obtain the value of $\alpha$ (in the figure) with an exponentially small error in polynomial time: we begin with an interval $[a, b] = [0, 1]$, fire the laser to $a$, $b$ and $\frac{a+b}{2}$ and decide in the following step to change one of $a$ or $b$ to $\frac{a+b}{2}$, based on which detectors reported a hit in each of the three firings. Should we couple this mechanical device to a Turing machine able to move the laser in this way and read from the detectors, we can decide exactly $\text{P/poly}$ in polynomial time.

4 HYPERCOMPUTERS

The most controversial view of hypercomputation pertains to the following:

**Thesis P (for ‘programmable’).** Hypercomputations can be specified for an apparatus which is not physically implausible.

There are two main paths leading to this thesis: Tien Kieu’s adiabatic quantum computer to solve Diophantine equations (18; 19) and Mark Hogarth’s Turing machines in curved space-time (15; 9; 35). Several objections have been put forward to Kieu’s approach (33; 14; 32; 11). Some have been answered (17; 21; 20), and the discussion is still active. Problems of a physical nature — related to blue shift — have been put forward against the use of curved space-time, which the authors believe to have solved (see, e.g., 23).

For more information, please consult the papers of Kieu and Hogarth in this special issue.

5 HYPERCOMPUTATION AS AN ORACLE OR ADVICE

The following thesis is also extremely debatable.

**Thesis O (for ‘oracle’).** The Universe has non-computable information content (which may be used as an oracle to build a hypercomputer).

\*In (3) the scatter machine is described as a Newtonian system: point particles replace the laser light and perfectly elastic barriers replace the mirrors. For this discussion the slight differences are irrelevant: we nevertheless mention that some issues which arise by the possibility of the point mass hitting the vertex of the barrier are solved by using mirrors and lasers.
Barry Cooper and Piergiorgio Odifreddi (5), for instance, have suggested similarities between the structure of the Universe and the structure of the Turing universe. Cristian Calude (4) investigates to what extent quantum randomness can be considered algorithmically random. The search for a physical oracle was proposed by Jack Copeland and Dianne Proudfoot (6). Their article and subsequent work have been severely criticised (7; 13) for historical and technical errors. There is, however, an appealing aesthetical side to what Copeland and Proudfoot proposed. The oracles of Turing machines are often seen and taught as abstract theoretical entities, merely technical devices. We nevertheless feel that it may be an interesting and beautiful endeavour to regard oracles as natural phenomena and study the oracles that arise in nature. We will give the example of Stonehenge, which will also illustrate what is meant by an oracle that arises in nature. The archaeological monument ‘Stonehenge’, located near Amesbury in the English county of Wiltshire, was built in three main phases from 3000BC to 1500BC. In the first main stage, called ‘Stonehenge I’, the monument was composed of around eighty standing stones. Fifty six of these stones were laid out in a circle around the centre of Stonehenge I (see Fig. 5), and are called Aubrey holes. We can number the Aubrey holes and make use of three tokens, placing each token in one of the holes: if the $i$th token is in the $n$th Aubrey hole, we say that the $i$th register holds the number $n$. In this peculiar way, Stonehenge I can be seen as a resource bounded implementation of a (Turing-universal) 3-counter machine. It is known, thanks to the work of Gerald Hawkins, Fred Hoyle and others (12; 16), that it is possible to use Stonehenge as a predictor of lunar and solar eclipses. From the point of view of the Earth both the Moon and the Sun follow approximately elliptical orbits, as shown in Fig. 4, which cross at the nodes N and N’. Suppose the moon is passing through N. Then a solar eclipse will occur if the sun is no further than 15° of N, and a lunar eclipse happens if the sun is within 10° of N’. If the moon is passing through N’ the situation is reversed. One can then wait for a solar eclipse, set the three tokens in the appropriate Aubrey hole, and use the following:

**Hoyle’s algorithm**

1. The first token, a little stone for instance, is moved along the Aubrey holes to keep track of the 28 day lunar cycle. We move the first token two places every day, since $56/2 = 28$.

2. The second token counts the days of the year. Since $56 \times 13/2 = 364$, we move the second token two places every thirteen days.
3. The third token will represent one of the nodes, say $N$. $N$ and $N'$ themselves rotate around the Earth, describing a full cycle (called a Metonic cycle) every 18.61 years. So we will move the third token three times every year, because $\frac{56}{3} = 18.67$.

4. Eclipses occur when the three tokens become aligned with each other up to one Aubrey hole to the right or to the left.

![Diagram of Earth, Moon, and Sun orbits with labels N, N', Lunar orbit, Solar orbit, Earth.](Image)

FIGURE 4
The approximate orbits of the Moon and the Sun around the Earth.

Ignoring the error for now, we conclude that simple modulo 56 arithmetic is enough to predict every eclipse with one single necessary input: the day of a solar eclipse when one sets the tokens in the first Aubrey hole. Now we introduce the Oracle. To the Northeast of Stonehenge I there is a 5 meter tall stone, called the ‘Heelstone’. In the morning of the Summer solstice the sun (our oracle) is born slightly to the north of the Heelstone. To know the exact day of the Summer solstice we wait for the day when the sun rises behind the Heelstone. The sunrise should then proceed north for a few days, and then back south. We count the number of days between the first sunrise behind the Heelstone and the second. The day of the summer solstice happened in the middle of these two events. With this information we can calibrate the sun token to enough precision every year, so that Stonehenge I can predict eclipses indefinitely.

---

§ The image in the first page illustrates where the sun would be, next to the heelstone, in the day of the summer solstice.

¶ The calibration procedure explained in (16) is slightly more complicated and detailed: we
We have described an unusual form of computation, aided by an unusual oracle. In our discussion we could have replaced Stonehenge I with a modern computer, and our oracle could be, for the sake of an example, a link with a satellite telescope. While it seems natural to consider the Sun as an oracle in the Stonehenge I algorithm described above, calling this satellite link an ‘oracle’ can feel awkward — one may prefer to call it ‘input’. We defend this decision by pointing out that apart from their symbolic value, these two sources of information have the same nature. It is customary to consider that input is finite, and given prior to the computation, but the sunrises or the satellite link give — in principle — an unbounded amount of data. They act as an oracle. Without these oracles both Stonehenge I and our modern computer would eventually be incapable of predicting eclipses, although the modern computer could keep providing accurate predictions for hundreds of years...

The remaining tokens can also be calibrated using other oracles: the phases of the moon give the adjustment of the first token and the precise day in which a solar eclipse occurs allows for calibration of the third token.
years.

Consider a variation of the Church–Turing thesis, opposite in nature to Thesis S. Such a variation could be phrased: the physical world is simulatable. This thesis leads us to conclude that one could, in principle, construct a Turing machine that could successfully predict eclipses forever, without the use of any oracle. Being able to predict eclipses indefinitely, however, would not imply that the physical world is simulable, unless the prediction of planet alignments (called conjunctions) is, in some sense, complete for the simulation problem.

6 HYPERCOMPUTATION AS A THEORY OF EVERYTHING

Penrose (24; 25) has put forward several arguments against the idea of a computable Universe. He distinguishes the cases where uncomputability manifests itself in essential and non-essential ways. Pour-el and Richard’s uncomputable solution of the wave equation is given as an example of the latter. Deeply rooted in Penrose’s work is a belief that the essential uncomputability of the Universe will provide evidence for:

Thesis E (for ‘everything’). The final theory of physics will be found to be uncomputable.

Physics, as a discipline, concerns the fundamental laws of the Universe, it aims to uncover the rules which govern the constituents of reality. The physicist may achieve, for this endeavour, two distinct levels of success: he may succeed at describing the laws, and he may additionally understand the rules. The physicist will have described the laws of the Universe if he has given a full account of, say, the equations which govern the elementary components of reality. He will have understood the rules of the Universe when he can, at least in principle, solve these equations and predict the behaviour of these components. Thesis E is the statement that one can describe the laws of the universe, but can not understand their essential mechanism. Penrose believes that the uncomputability described in this theory would aid in explaining the — seemingly non-algorithmic — phenomenon of conscious

---

# We feel it is justified, in this discussion, to abstract from the cosmological knowledge that eclipses will not happen forever.

** At this level of discussion we speak in principle, that is, we do not mean to say that physics only succeeds when it has entirely explained reality, but that this is physics Utopian goal. Speaking in these simplistic terms ignores several problems in the philosophy of science, but consciously so.
intelligence, and in this way contradict the strong A.I. thesis.

Please consult the paper by Apostolos Syropoulos in this issue for a discussion of related arguments.

7 CONCLUSION

We have pursued the three goals of surveying different models of hypercomputation, clarifying some misconceptions on the source of such computational power, and mapping out the research into five distinct trends. We have abstained from giving our opinion on these matters, by aiming for an informative rather than persuasive discussion of hypercomputation. We hope that by promoting a cleared understanding of the hypercomputational phenomena we have also stimulated a more relevant and informed criticism of hypercomputation, and hampered some misguided prejudice. Our division of hypercomputation into five views is possibly controversial. The advantage of having such a division is that each area, or trend, has a specific type of argument, or perspective, on hypercomputation. Understanding which arguments are required to defend each thesis, and which are the typical counter-arguments, should allow for a more focused, serious, and profound discussion of each hypercomputational model.

8 ACKNOWLEDGEMENTS

Bruno Loff would like to thank CMAF for the office space and the other fabulous resources made available to him. The authors are grateful to Mike Stannett for having invited them to submit this paper after the International Interdisciplinary Workshop on Future Trends in Hypercomputation, Sheffield, September 2006. José Félix Costa is also very indebted to the United Grand Lodge of England for providing him with the Stonehenge symbol. We both thank to the archaeologist Sir Gerald Hawkins and the (deceased) astronomer Sir Fred Hoyle. Finally, we thank the referees of this paper, for their detailed reading and excellent comments.

REFERENCES


