Computational complexity with experiments as oracles. II. Upper bounds

BY EDWIN BEGGS 1,*, JOSÉ FÉLIX COSTA 2,3, BRUNO LOFF 2,3 AND J. V. TUCKER 1

1 School of Physical Sciences, Swansea University, Swansea SA2 8PP, UK
2 Instituto Superior Técnico, Technical University of Lisbon, 1049-001 Lisbon, Portugal
3 Centro de Matemática e Aplicações Fundamentals, University of Lisbon, 1649-003 Lisbon, Portugal

Earlier, to explore the idea of combining physical experiments with algorithms, we introduced a new form of analogue–digital (AD) Turing machine. We examined in detail a case study where an experimental procedure, based on Newtonian kinematics, is used as an oracle with classes of Turing machines. The physical cost of oracle calls was counted and three forms of AD queries were studied, in which physical parameters can be set exactly and approximately. Here, in this sequel, we complete the classification of the computational power of these AD Turing machines and determine precisely what they can compute, using non-uniform complexity classes and probabilities.

Keywords: algorithmic procedure; experimental procedure; Turing machines with oracles; analogue–digital computation; non-uniform complexity; physical oracles

1. Introduction

In Beggs et al. (2008a), we introduced the idea of coupling a Turing machine to an abstract physical experiment as a new form of oracle. We focused on the design and operation of these computational models and the question: What can be computed by Turing machines with such physical oracles? In Beggs et al. (2008a), we chose a particular experiment and studied the hybrid machines made by combining Turing machines with the experiment.

Specifically, the experiment we chose was the scatter machine experiment (SME), which we had studied from a computational point of view in Beggs & Tucker (2007b). The SME is an experiment in Newtonian kinematics that measures to arbitrary precision the position $x$ of a vertex of a wedge, by firing particles and observing their behaviour. If the vertex position $x \in [0,1]$, then the SME can measure or compute rational approximations $x \approx \frac{p_n}{q_n}$ to any accuracy $|p_n - q_n| < 2^{-n}$, for $n = 1, 2, \ldots$. Thus, SME is a physical procedure that can
compute rational approximations of any real number in [0,1], which is a task far beyond the computational power of a Turing machine. Since the scatter machine can compute more than the Turing machine, it was an interesting experiment to choose as an oracle.

In Beggs et al. (2008a), the hybrid machines made by combining the SME and Turing machines were termed analogue–digital (AD) scatter machines. The case studies revealed some concepts and mathematical tools useful for a general theory of Turing machines with experimental oracles.

We introduced the concept of AD protocol for calls to the oracle to cover the complex process of

(i) setting parameters to the equipment,
(ii) performing the experiment, and
(iii) returning the results.

In particular, the AD protocol consumes resources, most obviously time: the cost of calling the oracle depends upon the size of the query and contributes to the complexity of computation. We use polynomial and exponential time AD protocols.

Furthermore, since the oracle is a physical process, we introduced three kinds of oracle query:

(i) **exact**, where a finite binary representation of numerical data \(z\) from the Turing machine sets a parameter of the experiment to an exact real number value \(r = z\),
(ii) **error-prone with unlimited precision**, where numerical data \(z\) from the Turing machine sets a parameter of the experiment to an approximate real number value \(r \in [z - \epsilon, z + \epsilon]\) for any requested \(\epsilon > 0\), and
(iii) **error-prone with fixed precision**, where numerical data \(z\) from the Turing machine sets a parameter of the experiment to an approximate real number value \(r \in [z - \epsilon, z + \epsilon]\) for an \(\epsilon > 0\) fixed for the machine.

With these concepts, we were able to prove theorems which partially answered the question about the computational power of the machines. Using the non-uniform complexity class \(P/\text{poly}\), we characterized the class of sets decidable by the polynomial-time computable exact AD scatter machines. Specifically, we proved the following for the error-free machines (Beggs et al. 2008a, theorems 7.2 and 7.4, respectively):

**Theorem 1.1.** The class of sets which are decidable in polynomial time by error-free deterministic AD scatter machines with a polynomial AD protocol is exactly \(P/\text{poly}\).

**Theorem 1.2.** The class of sets which are decidable in polynomial time by error-free deterministic AD scatter machines with a strictly exponential AD protocol is exactly \(P/\log^*\).

However, for the error-prone machines, we proved only lower bounds in Beggs et al. (2008a). In this paper, we answer questions left open in Beggs et al. (2008a) by proving upper bounds, which enable us to complete the classification of the
power of the three kinds of scatter machines. Combining upper and lower bounds, we formulate our new results as equivalence theorems. For the error-prone machines, we prove the following in §2.

**Theorem 1.3.** The class of sets which are decidable in polynomial time by error-prone arbitrary precision deterministic AD scatter machines with a polynomial AD protocol is exactly $P/poly$.

Thus, there is no difference in computational power between using exact queries and approximate queries having arbitrary precision.

In the proof of this equivalence, we need to be rather precise about just what we mean by ‘arbitrary precision’, specifically about how the variable size $\epsilon$ of the error is communicated from the Turing machine to the experiment. A scatter machine with arbitrary precision operates as follows. The word $z$ on the query tape represents the binary expansion of a dyadic rational and a particle is fired by a cannon that is placed somewhere in the accuracy interval $[z-2^{-|z|-1}, z+2^{-|z|-1}]$. This means that for different words representing the same dyadic rational, the longest word will give the highest precision. To increase the accuracy, it is only necessary to add more zeros to the binary representation of $z$. With this simple mechanism, we proved the lower bound in Beggs et al. (2008a). However, to prove the upper bound, we shall suppose that a uniform probability distribution governs the cannon position in this interval $[z-2^{-|z|-1}, z+2^{-|z|-1}]$.

The method of analysis and proof uses probabilistic Turing machines and so leads to the complexity class of $\text{BPP/}\text{poly}$ (which equals $\text{BPP/poly}$; see Balcazar & Hermo 1998). Now, this is known to be equal to $P/poly$. Previous appearances of $P/poly$ in models based on real numbers in the literature were discussed in Beggs et al. (2008a). However, in §3, we prove that the case of fixed precision is different.

**Theorem 1.4.** Assume an independent uniformly distributed cannon position in the accuracy interval. The class of sets which are decidable in polynomial time by error-prone fixed precision deterministic AD scatter machines is exactly $\text{BPP/}\text{log}^n$ independently of the complexity of the protocol.

Thus, using approximate queries having fixed finite precision results in a loss of computational power.

By fixed precision, we mean that the error is $2^{-N}$ for fixed $N$ and the accuracy interval is $[z-2^{-N}, z+2^{-N}]$. We assumed an independent uniformly distributed cannon position in the accuracy interval to prove the lower bound in Beggs et al. (2008a).

We could, of course, consider other fixed values, but we then have to consider the fact that the error chosen does affect the probabilities. To allow for the case when the error is not a dyadic rational, we could code the error into the advice and deal with a system with two physical constants (the error and the wedge position). But this sort of generality, at this stage, would...

---

1 The reasons why $\text{BPP/}\text{poly}=P/poly$ are as follows: on the one hand, for any two classes $A$ and $B$, if $A\subseteq B$, then $A/\text{poly}\subseteq B/\text{poly}$; thus, since $P\subseteq\text{BPP}$, we have that $P/\text{poly}\subseteq\text{BPP/}\text{poly}$; on the other hand, it is well known that $\text{BPP}$ has polynomial circuits, i.e. $\text{BPP}\subseteq P/\text{poly}$, and that $\text{poly}$ is idempotent; we deduce that $\text{BPP/}\text{poly}\subseteq P/\text{poly}/\text{poly}=P/\text{poly}$. See Balcazar et al. (1995).
only serve to obscure what is happening: our message is that we can compute with the system even if there is an absolute lower bound on the accuracy of queries.

The equivalence theorems here suggest that non-uniform complexity classes may have an important role to play in the exploration of the computational power of physical systems, a point we discuss in §4.

This paper is a sequel to Beggs et al. (2008a). We assume the reader is familiar with our earlier publication Beggs et al. (2008a) and so dispense with preliminaries that have been already summarized there. The essential topics are as follows: the definition and operation of the scatter machine (see §§3 and 5 in Beggs et al. 2008a), and the theory of oracle Turing machines, probabilistic Turing machines and non-uniform complexity classes (see Balcazar et al. 1995; §4 of Beggs et al. 2008a).

To appreciate fully the results in this paper, the reader will need to know the motivating problems, and the methodology that we have been developing to solve them, both of which are efficiently explained in Beggs & Tucker (2007b) and Beggs et al. (2008a). Fuller treatments of this background material can be found in Beggs & Tucker (2006, 2007a, 2008) and Beggs et al. (2008c).

2. Consulting the oracle with unlimited precision

In this section, we investigate the class of sets decidable in polynomial time by an error-prone AD scatter machine with arbitrary but finite precision and using a polynomial protocol. We will conclude that the computational power of these machines is not altered by considering such a small error, since they decide exactly $\text{BPP} \pm \text{poly} = \text{BPP}/\text{poly} = \text{P}/\text{poly}$ in polynomial time.

As in our former paper (Beggs et al. 2008a), we now show that only a number of digits of the vertex position, linear in the length of the computation, influence the decision of an error-prone AD scatter machine. Note that the behaviour of the error-prone AD scatter machines is probabilistic, because, if the machine shoots close enough to the vertex position, and with a large enough error $\epsilon$, the particle can go both left or right with a non-zero probability. Thus, we cannot ensure that, when we truncate the vertex position to $O(t(n))$ digits, the state of the machine will be exactly the same after $t(n)$ steps. Instead, we will show that if a machine decides in time $t(n)$, then by truncating the vertex position to $O(t(n))$ digits, the machine will decide the same set for inputs up to size $n$. In the following lemmas, remember that if a set is decided by an error-prone AD scatter machine with error probability bounded by $\gamma$, we may assume without loss of generality that $\gamma < 1/4$.

The whole computation can be summarized as follows. The Turing machine sets up an initial cannon position on the query tape. The experiment takes place and the result is either left or right. Owing to the error involved in setting up the actual physical position from the data on the query tape, the result of the experiment is given as a probability. Given the result of the experiment, the Turing machine sets a second cannon position and the process repeats. Given the bound on the time taken by the experiment, there can only be a certain number of experiments. Without loss of generality, it will be convenient to suppose that the number of experiments is always the same (we will justify this.
later). In figure 1, we show a tree describing the situation where there are three experiments. It is a binary tree as every experiment has two possible results, either left or right. At the end of the process, the Turing machine must either accept or reject, and this is shown by an ‘A’ or ‘R’ on each leaf of the tree. To obtain the probability of acceptance in figure 1, we take each descending path from the initial vertex, multiply the probabilities along its edges and add the results for each path ending in acceptance. Thus, the probability of acceptance in figure 1 is

\[
\frac{1}{3} \times \frac{1}{2} \times 0 + \frac{1}{3} \times \frac{1}{2} \times 1 + \frac{1}{3} \times \frac{1}{2} \times \frac{2}{3} + \frac{2}{3} \times \frac{1}{5} \times \frac{2}{3}.
\]

The reader should note that this tree, with its accept and reject states, is effectively ‘coded’ into, or specified by, the Turing machine, with the results of the experiments merely dictating which branch of the tree is followed at any stage. Each node or vertex of the tree could be labelled with the cannon position coded on the query tape if and when that experiment is performed. However, the probabilities attached to the branches are determined by the actual wedge position in the scatter machine. We shall study how the probabilities vary if we slightly change the wedge position. To do this, we shall develop some technical concepts and a useful proposition.

A probabilistic subtree will be a tree with probabilities attached to the edges as before, but now the probabilities may sum to less than 1. Thus, a subtree of figure 1 including the top node (the root) would be an example. Having the sum potentially less than 1 will be useful in a later proof by induction, but it also allows us to talk about (for example) a subtree ending in accepting states.

For a rooted tree \( T \), we denote by \( \mathcal{P}(T) \) the set of assignments of probabilities to the edges of \( T \), making \( T \) into a probabilistic subtree. Such an assignment is a map of the form \( D:E \to [0,1] \), where \( E \) is the set of edges of the tree \( T \). Then, for \( D \in \mathcal{P}(T) \), the total probability \( P(T, D) \) denotes the sum of the probabilities of each path from the root to the leaves of the tree \( T \). (The probability of a path is just the product of the probabilities in each of its edges.) In the case of figure 1, this is 1.

We denote the set of rooted trees of height \( \leq m \), having \( \leq 2 \) outgoing edges at every note, by \( T_m \).

Next, we define a distance between probabilistic subtrees on \( \mathcal{P}(T) \) by setting \( d(D, D') \) to be the maximum of the absolute values of the differences of the probabilities over the edges. The number \( d(D, D') \) is the largest deviation of the probabilities associated by \( D \) and \( D' \) to any edge.

**Figure 1.** A binary probabilistic tree of height 3.

```latex
\[
\frac{1}{3} \times \frac{1}{2} \times 0 + \frac{1}{3} \times \frac{1}{2} \times 1 + \frac{1}{3} \times \frac{1}{2} \times \frac{2}{3} + \frac{2}{3} \times \frac{1}{5} \times \frac{2}{3}.
\]
```
For example, the distance between the two trees in figure 2 is
\[
\max \left\{ \left| \frac{1}{2} - \frac{1}{5} \right|, \left| \frac{1}{2} - \frac{4}{5} \right|, \left| 0 - \frac{2}{3} \right|, \left| \frac{2}{3} - \frac{1}{3} \right|, \left| \frac{1}{2} - \frac{2}{3} \right| \right\}
\]
and their respective total probabilities are
\[
\frac{1}{2} \times 0 + \frac{1}{2} \times 1 + \frac{1}{2} \times \frac{1}{2} \quad \text{and} \quad \frac{1}{5} \times \frac{2}{3} + \frac{1}{5} \times \frac{1}{3} + \frac{4}{5} \times \frac{2}{3}.
\]

Now, with these concepts, we define a function \(f(m, k)\) for \(m \in \mathbb{N}\) and \(k \in [0,1]\) by
\[
f(m, k) = \sup_{T' \in T_m, D', D \in \mathcal{P}(T)} \sup_{d(D, D') \leq k} |P(T, D') - P(T, D)|.
\]

The function \(f(m, k)\) gives the largest possible difference in total probability for two different assignments of probabilities to a subtree of height \(\leq m\), where the distance between the probabilities is \(\leq k\). The function provides a measure that is uniform for trees of height up to \(m\) and assignments whose probabilities deviate up to \(k\).

**Proposition 2.1.** \(f(m, k) \leq 2mk\).

**Proof.** By induction on \(m\). The result is true for \(m=0\). Assume that it is true for \(m\) and consider \(f(m+1, k)\). A tree \(T \in T_{m+1}\) of height \(\geq 1\) can be written as one of the following, where \(X, Y \in T_m\):

\[
\begin{align*}
&\begin{array}{c}
\text{We shall only deal with the first case, the second (and easier case) is left to the reader. The tree can be assigned probabilities as follows, which we label} \\
D \in \mathcal{P}(T) \text{ and } D' \in \mathcal{P}(T), \text{ respectively:}
\end{array}
\end{align*}
\]

\[
\begin{align*}
&\begin{array}{c}
 p \quad q \\
(X, E) \quad (Y, F)
\end{array},
\end{align*}
\]

\[
\begin{align*}
&\begin{array}{c}
 p' \quad q' \\
(X, E') \quad (Y, F')
\end{array}.
\end{align*}
\]
Here, \( E, E' \in \mathcal{P}(X) \) and \( F, F' \in \mathcal{P}(Y) \). Now,
\[
P(T, D) = pP(X, E) + qP(Y, F), \quad P(T, D') = p'P(X, E') + q'P(Y, F').
\]
From this, we have
\[
|P(T, D') - P(T, D)| = p'(P(X, E') - P(X, E)) + q'(P(Y, F') - P(Y, F))
+ (p' - p)P(X, E) + (q' - q)P(Y, F),
\]
so we have
\[
|P(T, D') - P(T, D)| \leq p'f(m, k) + q'f(m, k) + |p' - p| + |q' - q|
\leq f(m, k) + 2k.
\]

**Lemma 2.2.** Let \( \mathcal{M} \) be an error-prone AD scatter machine with arbitrary precision, with the vertex at \( x \), deciding some set \( A \) in time \( t(n) \) with error probability bounded by \( \gamma \leq 1/4 \). Let \( \mathcal{M}' \) be an identical AD scatter machine, with the exception that the vertex is placed at \( x' \). If
\[
|x - x'| < \frac{2^{-t(n)}}{16t(n)},
\]
then for any word of size \( \leq n \), the probability of \( \mathcal{M}' \) making an error when deciding \( A \) is \( \leq 3/8 \).

**Proof.** We set the depth of the tree to \( t(n) \), as there may be at most \( t(n) \) experiments carried out in time \( t(n) \). The change in any particular probability is at most
\[
\frac{|x - x'|}{2\epsilon},
\]
where \( \epsilon \) is the error specified in the protocol. After carrying out \( t(n) \) steps or less, the size of the word representing \( z \), which was written in the query tape, is bounded by \( t(n) \), and thus the error in aiming the cannon is \( \epsilon = 2^{-t(n) - 1} \)—this is the smallest error possible. We then set
\[
k = 2^{t(n)} |x - x'|.
\]
For any particular word \( w \) of length \( |w| \leq n \), take the tree \( T \) to be the subtree of all possible experimental results which give the incorrect answer (this does not depend on the vertex position). Then, let \( D \in \mathcal{P}(T) \) be the probabilities on the edges given by the vertex at \( x \), and let \( D' \in \mathcal{P}(T) \) correspond to the vertex at \( x' \). The probability of an incorrect result for the wedge at \( x' \) is, by proposition 2.1,
\[
P(T, D') \leq P(T, D) + |P(T, D') - P(T, D)|
\leq \gamma + 2t(n)k
\leq \gamma + 2t(n)2^{t(n)} |x - x'|,
\]
so if we choose \( x' \) so that \( t(n)2^{t(n)} |x - x'| < 1/16 \), then \( P(T, D') < 3/8 \).
Lemma 2.3. Let $\mathcal{M}$ be an error-prone AD scatter machine with arbitrary precision, with the vertex at $x$, deciding some set in time $t(n)$ with error probability bounded by $\gamma < 1/4$. Let $\mathcal{M}_n$ be an identical AD scatter machine, with the exception that the vertex placed at $x|2z(n)+5$. Then, $\mathcal{M}_n$ decides the same set as $\mathcal{M}$, also in time $t(n)$, but with error probability bounded by $3/8$.

Proof. This is a corollary of lemma 2.2, using the fact that $t(n) \leq 2^{t(n)}$.

The reader should recall that the only important bit about the error probabilities is that the machine decides the set in polynomial time to an error strictly less than 1/2. Thus $\mathcal{M}$ and $\mathcal{M}_n$ above decide the same set for words of length $\leq n$. However, the important thing about $\mathcal{M}_n$ is that its wedge is positioned at a dyadic rational. Remember that we have also (by construction of the machine) a physical cannon position that is specified by a dyadic rational centre and width. Then, the probability of left or right is also a dyadic rational, and one which can be calculated given $x|2z(n)+5$ and the data on the query tape. This allows us to make a completely digital model of $\mathcal{M}_n$, provided that we have access to a fair coin-tossing machine to generate the dyadic rational probability. Following this line of reasoning gives the following result:

Theorem 2.4. Every set decided by an error-prone AD scatter machine with arbitrary precision in polynomial time is in BPP//poly. This is independent of the protocol.

Proof. Let $A$ be a set decided by an error-prone AD scatter machine $\mathcal{M}$ in polynomial time $p$ and with an error probability bounded by 1/4.

Let $x$ be the position of the vertex of $\mathcal{M}$. We use the advice function $f \in \text{poly}$, given by

$$f(n) = \tilde{x} = x|2^{p(n)+5},$$

(2.1)

to construct a probabilistic Turing machine $\tilde{\mathcal{M}}$ with advice which decides $A$. From lemma 2.3, we know that a machine with vertex at $f(n)$ will make the same decisions on words of length $\leq n$ as the one with the original vertex position, but with error probability bounded by 3/8. Now we will model the behaviour of the machine with dyadic rational vertex at $f(n)$ by using a program with access to a fair coin-tossing oracle.

Suppose that $\mathcal{M}$ is given a word of length $n$ as input. A call to the AD oracle in $\mathcal{M}$ consists of a word $z$ representing the binary expansion of a dyadic rational, and the cannon is placed according to a uniform probability distribution in the interval $[z - 2^{-|z|-1}, z + 2^{-|z|-1}]$. Considering the time taken to write the word to the query tape, we know that $|z| \leq p(n)$. If $f(n)$ is outside this interval (this takes a polynomial amount of time to check), we answer left or right with certainty (left if it is less than the lower bound and right if it is greater than the upper bound). If it is inside the interval, we use the fair coin-tossing oracle to simulate the uniform probability distribution for the cannon position. Using the fact that $\tilde{x} = f(n)$ is a rational with denominator $2^{2p(n)+5}$, we note that we obtain the following integer $m$ and accuracy interval:

$$m = 2^{2p(n)+5}(\tilde{x} - z + 2^{-|z|-1}) \in [0, 2^{2p(n)+5-|z|}].$$

In Beggs et al. (2008a), and throughout this paper, by $x \uparrow n$ we mean the $n$ first bits of $x$. 

Now make \( k = 2p(n) + 5 - |z| \) calls to the fair coin-tossing oracle, interpreting the results \( \tau_i \) (taken in order) as either 0 or 1. Now test if \( \tau_1 \tau_2 \cdots \tau_k < m \), where \( \tau_1 \tau_2 \cdots \tau_k \) is taken to be the binary representation of an integer. If the test is true, return \textit{right}, otherwise return \textit{left}. The probability of returning \textit{right} is \( m/2^k \), as required. The time taken is polynomial in \( n \).

In a run time of \( p(n) \), we can call the AD oracle in \( \mathcal{M} \) at most \( p(n) \) times, so the total run time of the simulated AD oracle machine \( \mathcal{M} \) is also polynomial.

In our paper (Beggs et al. 2008a), we proved the statement for lower bounds of computational complexity of the AD scatter machine with unbounded precision.

**Theorem 2.5.** Every set in \( \text{BPP}/\text{poly} \) can be decided in polynomial time by an error-prone AD scatter machine with arbitrary precision and polynomial protocol.

Combining the above theorems, the conclusion for this section is theorem 2.6.

**Theorem 2.6.** The class of sets decided by error-prone AD scatter machines with arbitrary precision and polynomial protocol in polynomial time is exactly \( \text{BPP}/\text{poly} = \text{P}/\text{poly} \).

### 3. Reducing the precision of the cannon

We will now study the class of sets decidable in polynomial time by an error-prone AD scatter machine with fixed precision. We will show that such machines may, in polynomial time, make probabilistic guesses of up to a logarithmic number of digits of the position of the vertex. We will also prove that these machines decide exactly \( \text{BPP}/\text{log}^* \).

The AD scatter machine combines a deterministic Turing machine with a physical oracle. When the oracle queries involve errors, the coupled machine behaves probabilistically. Specifically, our analysis used probability to handle the queries. We have seen that the computation tree of an AD scatter machine is deterministic except in the nodes that correspond to an oracle consultation. In Beggs et al. (2008a), we gave the lower bound theorem that any set in \( \text{BPP}/\text{log}^* \) can be decided in polynomial time by an error-prone AD scatter machine with fixed precision. Perhaps we left too many details to the reader, so here we take the opportunity to give a detailed proof before tackling the upper bound result. The reader is reminded that, so that we do not create the unnecessary complication of having to deal with two constants in the apparatus, we suppose that the fixed error is of the form \( \epsilon = 2^{-N} \).

Consider the class \( \text{BPP}/\text{log}^* \). A set \( A \) is in the class if we can find a suitable advice \( f \) of logarithmic size and a set \( B \) in \( \text{BPP} \), such that, for any input \( x \), \( x \in A \) if and only if \( \langle x, f(n) \rangle \in B \), for all \( n \geq |x| \). In Beggs et al. (2008a), the outline proof of the lower bound theorem concentrated on showing the following.

**Lemma 3.1.** An error-prone AD scatter machine with fixed precision, given input \( x \) of size \( |x| = n \), can read \( O(\log(n)) \) bits of the vertex position in polynomial time in \( n \), independently of the protocol.

If the advice \( f \) is coded as a vertex in the way described in Beggs et al. (2008a), then the lemma implies that the advice can be decoded and read in polynomial
time, no matter the protocol being polynomial or exponential. The reason for the independence is that the method uses repeated firings of the cannon from the same vertex position, so the cost in time is linear in the number of firings.

Next, we have to prove that given the input \( x \) and the advice \( f_j \), with the same AD scatter machine, we can show that \( x, f_j(x) \in B \). Moreover, since this in \( B \) is decided by a probabilistic Turing machine \( M \), we need to show that the AD scatter machine, although deterministic in its digital part, can simulate the probabilistic Turing machine. The proof uses the following, lemma 7.7 from Beggs et al. (2008a).

**Lemma 3.2.** Take a biased coin (probability of heads \( q \in (\delta, 1 - \delta) \) for some \( 1/2 > \delta > 0 \), and \( \gamma \in (0, 1) \). Then, up to probability \( \geq \gamma \), there is a sequence of independent coin tosses of length

\[
\frac{n}{\delta(1-\delta)} \left(1 + \frac{1}{\sqrt{1-\gamma}}\right),
\]

which gives a sequence of independent fair coin tosses of length \( n \).

**Theorem 3.3.** An error-prone AD scatter machine with fixed precision can decide any set in \( \text{BPP/}/\log^* \) in polynomial time, independently of the protocol.

**Proof.** Let \( A \) be an arbitrary set in \( \text{BPP/}/\log^* \) and \( M \) a probabilistic Turing machine with advice \( f \in \log^* \), which decides \( A \) in polynomial time with error probability bounded by \( \gamma < 1/7 \). Let \( a, b \in \mathbb{N} \) be such that \( |f(n)| = [a \log n + b] \).

We construct an error-prone AD scatter machine with fixed precision, \( \tilde{M} \), which estimates \( f(n) \) with an error probability bounded by \( 1/16 \), as in lemma 3.1. (We also make sure that the vertex position is estimated to within \( \epsilon/4 \), where \( \epsilon \) is the fixed error, again within the given error probability.) Now we need to simulate independent unbiased coin tosses, and we use lemma 3.2 to do this. If we can set the cannon to a position where the probability \( q \) of scattering left is in \( (\delta, 1 - \delta) \) for a known \( \delta > 0 \), then we have a known linear bound on the number of repeated firings we need to simulate a given number of independent unbiased coin tosses (to within a probability of \( 1/16 \)). If we set the cannon position to our estimate of the vertex position, we know that we can take \( \delta = 1/4 \), since we have estimated the vertex position to within \( \epsilon/4 \), where \( \epsilon \) is the fixed error. In fact, we do not need to use the full accuracy of our determination of the vertex position, we need only enough to give the accuracy within \( \epsilon/4 \), so we need only to set the cannon to a fixed number of binary places.

Our first result is similar to that of §2. We will show that the decision of an error-prone AD scatter machine with fixed precision remains the same if we keep only a number of digits of the vertex position logarithmic in the size of the input.

**Lemma 3.4.** Let \( M \) be an error-prone AD scatter machine with fixed precision \( \epsilon > 0 \), with the vertex at \( x \), deciding some set in time \( t(n) \) with error probability bounded by \( \gamma < 1/4 \). Let \( M' \) be an identical AD scatter machine, with the exception that the vertex is placed at \( x' \). If

\[
|x - x'| < \frac{\epsilon}{8t(n)},
\]

then for any word of size \( \leq n \), the probability of \( M' \) making an error is \( \leq 3/8 \). This result is independent of the protocol.
Proof. We set the depth of the tree to $t(n)$, as there may be at most $t(n)$ experiments carried out in time $t(n)$. The change in any particular probability is at most

$$k = \frac{|x-x'|}{2\varepsilon},$$

where $\varepsilon$ is the fixed error. For any particular word $w$ of length $|w| \leq n$, take the tree $T$ to be the subtree of all possible experimental results which give the incorrect answer (this does not depend on the vertex position). Then, let $D \in \mathcal{P}(T)$ be the probabilities on the edges given by the vertex at $x$, and let $D' \in \mathcal{P}(T)$ correspond to the vertex at $x'$. The probability of an incorrect result for the wedge at $x'$ is, by proposition 2.1,

$$P(T, D') \leq P(T, D) + |P(T, D') - P(T, D)|$$

$$\leq \gamma + 2t(n)k$$

$$\leq \gamma + t(n) \frac{|x-x'|}{\varepsilon}.$$ 

Lemma 3.5. Let $\mathcal{M}$ be an error-prone AD scatter machine with fixed precision $\varepsilon > 0$, with the vertex at $x$, deciding some set in time $t(n)$ with error probability bounded by $\gamma < 1/4$. Let $\mathcal{M}_n$ be an identical AD scatter machine, with the exception that the vertex is placed at $x|_{\log_2 t(n) - \log_2 \varepsilon + 5}$. Then, $\mathcal{M}_n$ decides the same set as $\mathcal{M}$, also in time $t(n)$, but with error probability bounded by $3/8$. This result is independent of the protocol.

Proof. This is a corollary of lemma 3.4.

Theorem 3.6. Every set decided by an error-prone AD scatter machine with fixed error (taken to be $2^{-N}$) in polynomial time is in $\text{BPP}/\text{poly}$.

Proof. The proof is in every way similar to the proof of theorem 2.4, using the advice function $f(n) = x|_{\log_2 t(n) - \log_2 \varepsilon + 5}$.

Theorem 3.7. The class of sets decided by error-prone AD scatter machines with fixed precision (taken to be $2^{-N}$) in polynomial time is exactly $\text{BPP}/\text{poly} = \text{P}/\text{poly}$.

Proof. One-half is theorem 3.3 (see also Beggs et al. 2008a) and the other half is theorem 3.6.

4. Conclusion

In our earlier publications (Beggs et al. 2008a–c), we began to explore the idea of combining physical experiments with algorithms, by using a physical process as an oracle to a Turing machine. We examined in detail case studies where an experimental procedure, the SME, based on Newtonian kinematics, is combined with a deterministic Turing machine. In Beggs et al. (2008a), we demonstrated that the notion had both conceptual and technical merits. The physical oracle leads directly to the new concepts of protocol and three forms of AD query, in which physical parameters can be set exactly and approximately. In particular, one characterization theorem and two lower bound theorems for computation by these AD machines were established using non-uniform complexity classes.
Here, in this sequel, we have completed the classification of the computational power of the two types of error-prone AD Turing machines, determining precisely what they can compute. Combining the lower bounds of Beggs et al. (2008a) with the upper bounds here, we have the following computational characterizations: for polynomial-time AD protocols,

\[ \text{AD scatter machines with infinite precision in polynomial time} \]
\[ = \text{AD scatter machines with finite but unlimited precision in polynomial time} \]
\[ = \text{BPP//poly} = \text{P/poly}, \]

but, using theorem 3.3, there is a reduction in power

\[ \text{AD scatter machines with finite fixed precision in polynomial time} = \text{BPP//log^*}. \]

The equivalence theorems tempt us to introduce classes for the sets computed by these new AD machines. Let \( E \) be some experiment that is suitable as an oracle. We can define

\[ TM(E_0)(f(n), g(n)), \quad TM(E_{\epsilon=0})(f(n), g(n)), \quad TM(E_\epsilon)(f(n), g(n)), \]

for Turing machines combined with the exact, error-prone with unlimited precision, and error-prone with fixed precision \( \epsilon \), experiment \( E \), operating with \( O(f(n)) \) computation time and \( O(g(n)) \) protocol time, respectively. However, at this early stage of development, several more experiments need to be studied to determine an appropriate theoretical framework.

The case \( E=\text{SME} \) is of general significance. We believe that the scatter machine is an important representative of a class of interesting experimental procedures: it has bounded space, linear time and energy and introduces bisection methods into experimental procedures. The equivalence theorems here suggest that non-uniform complexity classes have an essential role in understanding the computational capabilities of physical systems. Certainly, given that the power of the oracle, the SME, is provably beyond that of the Turing machine, it is fortunate that the theory of non-uniform classes exists and is able to analyse and capture this form of hypercomputation. Furthermore, in the proofs, the probabilistic Turing machines play an essential role in relating experiments and computers.

We believe that non-uniform complexity theory, and probabilistic algorithms, can classify the computations of a wide variety of combinations of different experimental and algorithmic procedures. This is supported by our new results on the scatter machine and other experiments. Different types of Turing machine can be coupled to the scatter machine and compared: in particular, the \( P=NP \) and similar problems can be relativized to the SME. More simply, the argument in theorem 3.3 of this paper can be developed to show the following.

**Theorem 4.1.** *Any probabilistic Turing machine working with time \( t(n) \) can be simulated by an error-prone AD scatter machine with fixed precision in time \( O(t(n)) \).*

*Proc. R. Soc. A (2009)*
In this paper, precision plays an important role. The role of protocols will be more prominent when combining Turing machines with a new experiment designed to measure mass using Newtonian collisions; this experiment in dynamics has an exponential protocol (see Beggs et al. in preparation).

The research of José Félix Costa is supported by Fundação para a Ciência e Tecnologia, Financiamento Base 2008—ISFL/1/209. E.J.B. and J.V.T. would like to thank EPSRC for their support under grant EP/C525361/1.

References


