COMPUTATIONAL CAPABILITIES OF PHYSICAL SYSTEMS

by David H. Wolpert

NASA Ames Research Center, N269-1, Moffett Field, CA 94035, dhw@ptolemy.arc.nasa.gov

PACS numbers: 02.10.By, 02.60.Cb, 03.65.Bz

Abstract: In this paper strong limits on the accuracy of real-world physical computation are established. To derive these results a non-Turing Machine (TM) formulation of physical computation is used. First it is proven that there cannot be a physical computer C to which one can pose any and all computational tasks concerning the physical universe. Next it is proven that no physical computer C can correctly carry out every computational task in the subset of such tasks that could potentially be posed to C. This means in particular that there cannot be a physical computer that can be assured of correctly "processing information faster than the universe does". Because this result holds independent of how or if the computer is physically coupled to the rest of the universe, it also means that there cannot exist an infallible, general-purpose observation apparatus, nor an infallible, general-purpose control apparatus. These results do not rely on systems that are infinite, and/or non-classical, and/or obey chaotic dynamics. They also hold even if one could use an infinitely fast, infinitely dense computer, with computational powers greater than that of a Turing Machine (TM). After deriving these results analogues of the TM Halting theorem are derived for the novel kind of computer considered in this paper, as are results concerning the (im)possibility of certain kinds of error-correcting codes. In addition, an analogue of algorithmic information complexity, "prediction complexity", is elaborated. A task-independent bound is derived on how much the prediction complexity of a computational task can differ for two different reference universal physical computers used to solve that task. This is analogous to the "encoding" bound governing how much the algorithm information complexity of a TM calculation can differ for two reference universal TMs. It is proven that either the Hamiltonian of our universe proscribes a certain type of computation, or prediction complexity is unique (unlike algorithmic information complexity). Finally, the implications of this analysis for the issue of whether the universe "is" a computer are briefly discussed.

INTRODUCTION

Recently there has been heightened interest in the relationship between physics and computation ([1-37]). This interest extends far beyond the topic of quantum computation. On the one hand, physics has been used to investigate the limits on computation imposed by operating computers in the real physical universe. Conversely, there has been speculation concerning the limits imposed on the physical universe (or at least imposed on our models of the physical universe) by the need for the universe to process information, as computers do.

To investigate this second issue one would like to know what fundamental distinctions, if any, there are between the physical universe and a physical computer. To address this issue this paper begins by establishing that the universe cannot contain a computer to which one can pose any arbitrary computational task. Accordingly, this paper goes on to consider computer-indexed subsets of computational tasks, where all the members of any such subset *can* be posed to the associated computer. Restricting attention to such subsets, it then proves that one cannot build a computer that can "process information faster than the universe". More precisely, it is shown that one cannot build a computer that can, for any physical system, correctly predict any aspect of that system's future state before that future state actually occurs.

This asymmetry in computational speeds constitutes a fundamental distinction between the universe and the set of all physical computers. Its existence casts an interesting light on the ideas of Fredkin, Landauer and others concerning whether the universe "is" a computer, whether there are "information-processing restrictions" on the laws of physics, etc. [11, 20]. In a certain sense, the universe is more powerful than any information-processing system constructed within it could be. This result can alternatively be viewed as a restriction on the computational power of the universe — the universe cannot support the existence within it a computer that can process information as fast as it can.

To establish this unpredictability result this paper considers a model of physical computation that is actually general enough to address the performance of other computational tasks as well as prediction of the future. In particular, this model does not rely on temporal orderings of events, and therefore the unpredictability results also establish that no computer can infallibly predict the *past* (i.e., perform retrodiction). So any memory system must be fallible, i.e., the second law cannot be used to ensure perfectly faultless memory of the past. (Accordingly, the psychological arrow of time is not inviolate [31].[1]) The unpredictability results are also general enough to allow arbitrary coupling of the computer and the external universe. So for example they also establish that there cannot be either an infallible general purpose observation device nor an infallible general purpose control device. (The result concerning observation can be viewed as an uncertainty principle, one that does not involve quantum mechanics.)

No physically unrealizable systems, chaotic dynamics, or non-classical dynamics are exploited in this paper, and the results hold even if one restricts attention to predicting systems that contain a finite number of degrees of freedom. The results also hold even if the computer is infinitely dense and/or infinitely fast, even if the computer has an infinite amount of time to do the calculation (either before or after the event being predicted occurs). The results also hold even if the computer's initial input explicitly contains the correct value of the variable it is trying to predict / observe. More generally they hold regardless of the program running on the computer. They also hold for both analog and digital computation, and whether or not the computer's program can be loaded into its own input (i.e., regardless of the computational universality of the computer). In fact they hold regardless of the (Chomsky hierarchy) power of one's computer, so long as it is physically realizable. If it turns out to be physically possible to have computers with computational power greater than that of a Turing machine, then the result of this paper holds for such a computer. As a particular example, the results also hold even if the "computer" includes one or more human beings. So even if Penrose's musing on quantum gravity and intelligence turns out to be valid — even if human computational powers are not subject to the restrictions that apply to any of the members of the Chomsky hierarchy — it is still true that human intelligence is *guaranteed* to be wrong sometimes.

Results of such generality are derived by examining the underlying issues from the perspective of the computational character of real-world physical systems in general, rather than that of some single precisely specified (and often non-physically realizable) computer system. The associated mathematics does not directly involve dynamical systems like Turing machines. Rather it casts computation in terms of partitions of the space of possible worldlines of the universe. For example, to specify what input a particular physical computer has at a particular time is to specify a particular subset of all possible worldlines of the universe; different inputs to the computation correspond to different (non-overlapping) such subsets. Similar partitions specify outputs of a physical computer. Results concerning the (im)possibility of certain kinds of physical computation are derived by considering the relationship between these kinds of partitions. In its being defined in terms of such partitions, "physical computation" involves a structure that need not even be instantiated in some particular physically localized apparatus; the formal definition of a physical computer is general enough to also include more subtle non-localized dynamical processes unfolding across the entire universe. Computers in the conventional, space-time localized sense (e.g., the box on your desk) are simply special examples, with lots of extra restrictions that turn out to be unnecessary in the underlying mathematics.

Section 1 of this paper generalizes from particular instances of real-world physical computers that "try to reliably and ahead of time predict the future state of any system" to motivate a broad formal definition of physical computation in terms of partitions. To maintain maximum breadth of the analysis, we do not want to restrict attention to physical computers that are (or are not) capable of self-reference. As an alternative, we start by restricting attention to universes containing at least two physical computers. (Put another way, our initial results hold for any single computer not so powerful as to preclude the possible existence anywhere else in the universe of another computer as powerful as it is — which certainly describes any computer that human beings can ever create.) Section 1 also establishes that there exist prediction problems that cannot even be posed to one of those two physical computers. Restrictions on the set of prediction problems are introduced accordingly.

Section 2 proves that, even within such a restricted set of prediction problems, one cannot have a pair of computers each of which can, reliably and ahead of time, predict the future state of any system. It is also in Section 2 that the impossibility of an infallible general-purpose retrodiction apparatus, observation apparatus, or control apparatus is established. These results are all derived through what is essentially a physical version of a Cretan Liar's paradox; they can be viewed as a physical analogue of Godel's Incompleteness Theorem, involving two instances of the putative computer rather than self-referential computers.

The mathematics and impossibility results governing the partitions underlying computation bear many parallels with that governing conventional computer science models. Section 3 explicates some of that mathematical structure, involving topics ranging from error correction to the (lack of) transitivity of computational predictability across multiple distinct computers. In particular, results are presented concerning physical computation analogues of the mathematics of Turing machines, e.g., "universal" physical computers and Halting theorems for physical computers. In addition, an analogue of algorithmic information complexity, "prediction complexity", is elaborated. A task-independent bound is derived on how much the prediction complexity of a computational task can differ for two different reference universal physical computers used to solve that task. This bound is similar to the "encoding" bound governing how much the algorithmic information complexity of a Turing machine calculation can differ for two reference universal Turing machines. It is then proven that one of two cases must hold. One is that the Hamiltonian of our universe proscribes a certain type of computation. The other possibility is that, unlike conventional algorithmic information complexity, its physical computation analogue is unique, in that there is one and only version of it that can be applicable throughout our universe.

Section 4 presents a brief overview of how, the unpredictability results notwithstanding, this paper's formalism might be used to gainfully view a universe as a (single) computer. The implications of this paper's results under such an identification are briefly discussed. This section then relates the work presented in this paper to previous work in the literature, and ends with a discussion of future work.

Throughout this paper, $\mathbf{B} \equiv \{0, 1\}$, $\Re$ is defined to be the set of all real numbers, '^' is the logical *and* operator, and 'NOT' is the logical *not* operator applied to **B**. To avoid proliferation of symbols, often set-delineating curly brackets will be used surrounding a single symbol, in which case that symbol is to taken to be a variable with the indicated set being the set of all values of that variable. So for example "{y}" refers to the set of all values of the variable y. In addition o(A) is the (potentially transfinite) cardinality of any set A, and 2^A is the power set of A. $u \in U$ are the possible states of the universe, and $\hat{U}$ is the space of allowed trajectories through U (i.e., world-lines of the universe). So $\hat{u} \in \hat{U}$ is a single-valued map from $t \in \Re$ to $u \in U$, with $u_t \equiv \hat{u}_t$ the state of the universe at time t. Note that since the universe is microscopically deterministic, (be it classical or quantum-mechanical, if we adopt the many-worlds interpretation for the latter case) u_t for any t uniquely specifies $\hat{u}$. Sometimes there will be implicit constraints on $\hat{U}$. For example, we will assume in discussing any particular computer that the space $\hat{U}$ is restricted to worldlines $\hat{u}$ that contain that computer. Fully formal definitions and proofs are relegated to the appendix, so that the main text can concentrate on the fundamental concepts. Extra discussion and examples of those concepts that would be too distracting in the main text are also presented in the appendix. An earlier analysis addressing some of the issues considered in this paper can be found in [33].

I. A DEFINITION OF WHAT IT MEANS TO "PREDICT THE FUTURE"

i) Definition of a Physical Computer

For the purposes of this paper, a physical computer will "predict the state of a system ahead of time" if the computer is a general emulator of the physical dynamics of such a system, an emulator that operates faster than that dynamics. So given some time T > 0, and given some desired information concerning the state of some system at T, our goal is to have the computer output that desired information *before time T.* To that end we allow the computer to be "initialized" at time 0, with different "input", depending on the value of T, what information is desired, perhaps information about the current state of the state whose future is being predicted, etc.

To make this concrete, let α be a characteristic of the state of the physical universe at time T.

We indicate a specification that we wish to know α as a **question** $q \in Q$. So q says what α is for any state of the universe at time T, i.e., q is a single-valued mapping from the state of the universe at T to an **answer** α.

Since $\hat{u}$ fixes u_T and (for a deterministic universe) vice-versa, we can generalize this by dispensing with specification of T. In other words, we can recast any q as any single-valued mapping from $\hat{u}$ to α. So q fixes a partition over the space $\hat{U}$, and any pair (α, q) delineates a region in $\hat{U}$.

In general, the space $\{\alpha\}$ of potential answers of the universe (i.e., the set of partition element labels) can change depending on q, the question concerning the universe (i.e., the partition). This means that we need to concern ourselves not just with the relation between computers' answer values, but also with the relation between the associated spaces of possible values (e.g., the number 1 is both an element of the space **B** and of the space $\{1,4,5\}$, two cases that must be distinguished.) We will write the space $\{\alpha\}$ as A(q) when we need to indicate its dependence on q explicitly. As much as possible, the extra complexity associated with keeping track of A(q) is relegated to the fully formal analysis in the appendix.

Without the accompanying q, a value of α, by itself, is meaningless. So we must know what q we are answering when we read the computer's output. Accordingly, we want the output of our computer to give a question q together with an associated prediction for α. So our computer's output provides a delineation of a subregion of $\hat{u} \in \hat{U}$; those $\hat{u}$ such that $q(\hat{u}) = \alpha$. It provides more structure than just that though, e.g., two different outputs can have the same answer even though they delineate different regions (due to having different questions).

Very often the question — a mapping from answers to associated sets of possible states of the real world — is only stored in a human user's memory. In this case that aspect of the human is implicitly part of the computer. An example is where some particular pattern of bits in an electronic workstation needs to be "interpreted" by a human to serve as a prediction concerning the physical universe.

Finally, choose some real number τ, where $0 < \tau < T$. Our goal is that for any $q \in Q$ there is an associated initial "input" state of the computer at time 0 which ensures that at time τ our computer's output is a correct prediction for α, i.e., which ensures that for the $\hat{u}$ of the universe, $q(\hat{u}) = \alpha$.

Note that despite the nomenclature, a "question / answer" pair is not a premise and associated conclusion, in the sense of an if-then statement. Rather it is just a conclusion. The associated premise (i.e., the if clause) is encoded in the input.

Now consider in more detail a conventional computer that consists of a fixed physical dynamical system. Together with that system we have a pair of mappings by which some of that system's observable degrees of freedom are interpreted as (perhaps binary) "inputs", and some as "outputs". The input and output degrees of freedom can overlap, and may even be identical. Since the computer exists in the physical universe its state at any moment t is specified by u_t. Therefore both the interpretation of some of the computer's degrees of freedom as "inputs" and some as "outputs" are single-valued mappings from $u \in U$ to a space of inputs and of outputs, respectively. With the input time 0 and output time τ implicit, we can recast the domains of those mappings as $\hat{U}$ rather than U.

All of this holds whether the computation of outputs from inputs proceeds in a "digital" or "analog" fashion. The only restriction will be that we are interested in falsifiable rather than probabilistic predictions. This restriction will often be met even if the system being predicted is stochastic and the precise aspect of it we're predicting is a function of the associated distributions. For example, whether the temperature of a particular system falls within a certain range at a particular time is a falsifiable prediction. (See also Ex. 1 below.) In any case, the extension to having the computer's output be a probability distribution is fairly straight-forward — see the discussion just before Thm. 2.

Example 1 (conventional prediction of the future): Say that our universe contains a system S external to our computer that is closed in the time interval [0, T], and let u be the values of the elements of a set of canonical variables describing the universe. α is the t = T values of the components of u that concern S, measured on some finite grid $G(U_T)$. q is this definition of α with G and the like fully specified. (So q is a partition of the space of possible u_T, and α is an element of that partition.) Q is a set of such q's, differing in G, whose associated answers our computer can (we hope) predict correctly. By determinism, under the convention that we are interested in questions concerning the t = T state of the universe, we can replace any grid $G(U_T)$ with a grid $G(\hat{U})$.

The input to the computer is implicitly reflected in its t = 0 physical state, as our interpretation of that state. In this example (though not necessarily in general), that input specifies what question we want answered, i.e., which q and associated T we are interested in. It also delineates one of several regions $R \subseteq \hat{U}$, each of which, intuitively, gives the t = 0 state of S and S's Hamiltonian. Throughout each such R, the system S is closed from the rest of the universe during $t \in [0, T]$. Since the precise R delineated specifies a set of possible values of u_0 in full, not just of S's t = 0 state, it is an element of a (perhaps irregular) finite precision grid over $\hat{U}$, G'. If, for some R, $q(\hat{u})$ has the same value for all $\hat{u} \in R$, then this input R uniquely specifies what α is for any associated $\hat{u}$. If this is not the case, then the R input to the computer does not suffice to answer question q. So for any q and region R both of which can be specified in the computer's input, R must be a subset of a region $q^{-1}(\alpha)$ for some α.

Implicit in this definition is some means for correctly getting the information of the value R into the computer's input. In practice, this is often done by having had the computer coupled to S sometime before time 0. As an alternative, rather than specify R in the input, we could have the input contain a “pointer” telling the computer where to look to get the information R. (The analysis of this paper holds no matter how the computer gains access to R.)

In practice the input, giving R, q, and T, is an element of a partition over an “input section” of our computer. In such a case, the input is itself an element of a finite precision grid over U_0, $G''(U_0)$. So an element of G" specifies an element of G (namely q) and element of G' (namely R).

As usual any $G''(U_0)$ can be re-expressed as a grid $G''(\hat{U})$, under the convention that we are interested in inputs imposed on the t = 0 state of the computer. Note that if initialization were to be at a time $t \neq 0$, it would correspond to a different grid $G''(\hat{U})$, in general, since the values of the computer's input degrees of freedom may vary in time.

Given its input, the computer (tries to) form its prediction for α by first running the laws of physics on a u_0 having the specified value as measured on G', according to the specified Hamiltonian, up to the specified time T. The computer then applies q(.) to the result. Finally, it ensures that this prediction for α is in its output section at time τ. More precisely, there is a fourth finite precision grid G''' over U_τ defined by the state of the computer's output section at time τ. The computer uses that grid to "write out" (what is interpreted as) its prediction for which region in U the universe will be in at T, that prediction being formally equivalent to a prediction of a region in $\hat{U}$. The goal is to have it do this, with the correct value of α, by time $\tau < T$.

Since G'''(U_τ) induces a grid over $\hat{U}$, G'''($\hat{U}$), we can dispense with the "time $\tau \le T$" stipulation; the goal is simply to have the universe be in the element of G'''($\hat{U}$) associated with the correct value of α. As with changing the time of input, changing the time τ of output will change the grid G'''($\hat{U}$), in general.

Consider again the case where there is in fact a correct prediction, i.e., where R is indeed a subset of the region $q^{-1}(\alpha)$ for some α. For this case, formally speaking, "all the computer has to do" in making its prediction is take the region R and question q delineated in its input and recognize which region in the partition q contains the region R. Then it must output the label of that region in q onto its output. In practice though, q and R are usually "encoded" differently, and the computer must "translate" between those encodings to recognize which region $q^{-1}(\alpha)$ contains R; this translation constitutes the "computation".

Note that all of this holds even if S's dynamics is stochastic, and/or S's state is never deterministically fixed to greater precision that that of G'.

Generalizing these considerations, we define a computer's **input** to be a mapping X(.) from $\hat{u} \in \hat{U}$ to a space of inputs, $\{x\}$. Intuitively, it is a partition of $\hat{U}$ (see the appendix). So for example "initialization" of a computer as conventionally conceived, which sets the t = 0 state of a physical system underlying the computer, is simply a special case. (In that special case, the value taken by the input mapping differs for $\hat{u}$ and $\hat{u}'$ if the t = 0 state of the computer input portion of the universe, as specified by $\hat{u}$, differs from the t = 0 state of the computer input portion of the universe as specified by $\hat{u}'$.) Similarly, we can define a computer's **output** to be a mapping Y(.) from $\hat{u} \in \hat{U}$ to a space of outputs, {y}. In such an output partition, the set {y} consist of all pairs $\{y_q \in Q, y_\alpha \in A(y_q)\}$, for some Q and associated A(.). We say that y_q is the "question posed to the computer", and y_α is "the computer's answer".

A **physical computer** then is simply the double of an input partition and an associated output partition. As considered in this paper, all that computation amounts to is the delineation of the logical implications for which element(s) of the output partition contain $\hat{u}$, given that a particular provided input partition element contains $\hat{u}$. We are interested in whether the element of the output partition induced by a particular input correctly describes the universe, as restricted by that input. So in particular, we are not considering counter-factual "computation" involving premises that conflict with the actual state of the universe.

The definition of a physical computer presented here is far broader than conventional computers that work by processes like that outlined in Ex. 1, as the following discussion explicates:

Example 1 continued: The definition of a physical computer does not require that an input always implies a unique output, as in Ex. 1. In addition, the computer in Ex. 1 has the laws of physics explicitly built into its "program". But our definition allows arbitrary "programs". Our definition also allows other kinds of information input to the computer besides that of Ex. 1. Furthermore, we will only need to require that there be *some* input to the computer that, by accident or by design, induces the correct output. This means we do not even require that the computer's initial state x "accurately describes" the t = 0 external universe in any meaningful sense. (Our generalization of Ex. 1 preserves analogues of the grids G (in Q(.)), G'' (in X(.)) and G''' (in Y(.)), but not of the grid G'.)

In fact, since the partition X(.) can reflect *any* attribute of $\hat{u}$, it need not even involve the t = 0 state of the physical computer. In other words, as we use the terms here, the computer's "input" need not be specified in some t = 0 state of a physical device. Indeed, our definition does not even explicitly delineate the particular physical system within the universe that we identify with the computer. (A physical computer is simply an input partition together with an output partition.) This means we can even choose to have the entire universe "be the computer" (see Sect. 4). In addition, our definition does not enforce having inputs be "set" before outputs are "read" in any sense. It is only concerned with the entire worldlines of the universe.

As another example of the freedom to extend Ex. 1, note that in practice we may want to physically couple our computer to the external universe, for example via an observation apparatus that initializes the computer's inputs so that they reflect information about the system being predicted. Such a coupling would be reflected in $\hat{u}$. If we wish though, we can exploit the freedom in its definition to modify the input mapping, in such a way that it too directly reflects this kind of coupling. For example, under the proposed modification, if we want the input section of the computer's underlying physical system to be a bit b_1 that equals the t = -1 state of some bit b_2 concerning the external universe, then we could have $X(\hat{u}) = X(b_1(u_0), b_2(u_{-1})) = b_1(u_0)$ if $b_1(u_0) = b_2(u_{-1})$, and have it equal a special "input error" value otherwise. If we do have a physical coupling mechanism, and if that mechanism is reliable — something reflected in $\hat{u}$ — then this third setting will never occur, and we can ignore it. However use of this modified X allows us to avoid explicitly identifying such a mechanism and simply presume its existence. So long as the third setting never occurs, we can analyze the system *as though* it had such a (reliable) physical coupling mechanism.

We can also modify Ex. 1 in other ways that do not involve input. For example, we can have S be open (or perhaps even be the entire universe). We can also have the computer observe the system being predicted *after* initialization (so that that initialization only serves to specify what should be observed). This is one of the major reasons why we do not require that the value x uniquely fixes $Y_\alpha(\hat{u})$, to not preclude the possibility of y_α being based on observations of the external world that occur after the setting of the computer's input. (Other reasons for not having x fix y_α arise in the context of weak predictability; see the discussion in the appendix preceding Ex. 2.) Other examples of how to modify Ex. 1 are presented below in the discussion of retrodiction and control.

We will sometimes find it useful to consider a **copy** of a particular computer C = (X, Y). This is any computer C' = (X', Y') where {x'} = {x}, {Y'} = {Y}, and the (set-valued) function of all outputs that are possible given a particular input is the same for both computers. In other words, even though the functions X'(.) and Y'(.) may differ from X(.) and Y(.) respectively, the logical implications relating values of x' and Y' are the same as those relating values x and Y. So both computers have the same input-output mapping. As a particular example, if a scientist at a particular time (i.e., a computer) C in some space $\hat{U}$ is transformed into a copy C' in some $\hat{U}'$, there is no way that (s)he can ascertain that that transformation has occurred. The two scientists interpret their input as the same question and in response provide the same answer (whether that answer is generated via prediction and/or observation — see the discussion below Thm. 2).

Example 1 continued: Consider again the computer in Ex. 1. Recall that if the initialization time 0, question time T, and/or output time τ are changed, then in general the partitions X and/or Y may change. So in particular, the time-translated version of a computer C differs from C, in general. However the "time-translated version of C" is a copy of C (or at least it makes sense to define the term that way, so long as the laws of physics are time-translation invariant). Similarly, a spatially-translated version of C is only a copy of C in general, rather than identically equal to C. So formally speaking, the sequence of computations the box on your desk makes over a period of a month is a set of physical computers, all copies of one another, applied to the same $\hat{u}$.

ii) Intelligible computation and distinguishable computers

Consider a conventional physical computer, consisting of an underlying physical system whose t = 0 state sets X($\hat{u}$) and whose state at time τ sets Y($\hat{u}$), as in Ex. 1. We wish to analyze whether the physical system underlying that computer can calculate the future sufficiently quickly. In doing so, we do not want to allow any of the "computational load" of the calculation to be "hidden" in the mappings X(.) and Y(.) by which we interpret the underlying physical system's state, thereby lessening the computational load on that underlying physical system. Stated differently, we wish both the input and the output corresponding to any state of the underlying physical system to be "immediately and readily intelligible", rather than requiring non-trivial subsequent computing before it can be interpreted. As will be seen in our formalization of this requirement, it is equivalent to stipulating that our computer be flexible enough that there are no restrictions on the possible questions one can pose to it.

One way to formalize this intelligibility constraint would entail imposing capabilities for self-reference onto our computer. This has the major disadvantage of restricting the set of physical computers under consideration. As an alternative, to formalize the notion that a computer's inputs and outputs be "intelligible", here we consider universes having another computer which can consider the first one. We then require that that second computer be able to directly pose binary questions about whether the first computer's prediction correctly corresponds to reality, without relying on any intervening "translational" computer to interpret that first computer. (Note that nothing is being said about whether such a question can be correctly *answered* by the second computer, simply whether it can be *posed* to that computer.) So we wish to be able to ask if that output is one particular value, whether it is another particular value, whether it is one of a certain set of values, etc. Intuitively, this means that the set Q for the second computer must contain binary functions of Y(.) of the first computer. Finally, we also require that the second computer be similarly intelligible to the first one.

These two requirements are how we impose the intuitive requirement that both computers be "readily intelligible" as predictions concerning reality; they must be readily intelligible and checkable *to each other*. More precisely, define an **intelligibility function** of any $\hat{U}$-partition π to be a binary-valued function of the elements of that partition. (We call a set of such functions an **intelligibility set**.) If the set of questions we can pose to a computer C includes all such functions, we say that π is **intelligible** to C. For such a case, C can have posed any question concerning the universe as measured on π. This flexibility in C ensures that C's output partition isn't "rigged ahead of time" in favor of some particular question concerning π. The obvious modifications are assumed if we talk about π being intelligible to C "with respect to some intelligibility set F".[2]

A problem with this definition of intelligibility is that one can prove there cannot be a computer to which one can even pose all possible questions concerning the physical world. (This is established formally as Thm. 1 in the appendix.) The problem arises when we try to pose intelligibility functions concerning the computer C's output partition to C itself. Intuitively, it is not possible for the set of C's question partitions to include the (larger) set of all binary-valued functions of those partitions.

To circumvent this problem, from now on we implicitly restrict any intelligibility function concerning an output partition Y to be **question-independent**, i.e., to not depend on the precise question encoded in y, only on the answer component. Intuitively, restricting ourselves to these kinds of intelligibility functions means we are only requiring that the predicted partition *label* of one physical computer be directly readable on the other computer's input, not that the full partition of the first computer's question also be directly readable. Given the restriction to such question-independent intelligibility functions, we say that two physical computers C^1 and C^2 are **mutually intelligible** if the output partition of C^2 is intelligible to C^1 and vice-versa.

Formally speaking, to make sure that the ranges of intelligibility functions match up with those of output partitions when working with question-independent intelligibility functions, often we should consider the full **p**rediction partition, $Y_p(\hat{u}) \equiv (A(Y_q(\hat{u}), Y_\alpha(\hat{u}))$, rather than just $Y_\alpha(\hat{u})$. For example, this is the case in the formal definitions of weak and strong predictability (see the appendix). For pedagogical simplicity though, we will often just refer to the "computer's answer" or the "computer's prediction" rather than explicitly state whether we mean Y_p. As always, such formal concerns are dealt with in full in the appendix.

Finally, our unpredictability results will rely on our two physical computers being distinct from one another. They must not be so intertwined that how we can initialize one of them is determined by how we initialize the other. More formally, just as we require that all input values $x \in$

{x} are physically realizable states of a single physical computer, so all pairs of the two computer's inputs values must be physically realizable states of the two physical computers. When this is the case we say that the computers are **pairwise (input) distinguishable**. When this is the case for each pair of a set of computers, we say that the set is pairwise-distinguishable, and when it is possible to have any joint combination of the input values of all members of the set we say we have **full distinguishability** for that set.

iii) Predictable computation

We can now formalize the concept of a physical computer's "making a correct prediction" concerning another computer's future state. We say that a $\hat{U}$-partition π is **weakly predictable** to C if two conditions hold. First, π must be intelligible to C. Second, for every intelligibility function concerning π, f, $\exists$ x $\in$ {x} that **weakly induces** f, i.e., a value x such that $X(\hat{u}) = x$ forces the prediction to equal $f(\hat{u})$. We will say a computer C' with output Y'(.) is weakly predictable to another computer C, and write C > C', if the answer partition of C' is weakly predictable to C. If we just say "predictable" it will be assumed that we mean weak predictability.

See the variants of Ex. 2 in the appendix for illustrations of weakly predictable sets of computers. These demonstrate, among other things, that the ">" relation need not be transitive. In fact, even if some C^1 could predict C^2's input simultaneously with predicting C^2's answer, it still would not follow that C^1 can predict some π just because C^2 can. This is because C^1 has no ability to set its input to ensure that x^2 is one of the values involved in C^2's predicting π. (Strong predictability, introduced below, rectifies this.)

This definition of predictable is very broad. It doesn't require that there be a sense in which the information input to C is interpretable as a description of the external universe. (This freedom is what allows us to avoid formalizing the concept of whether some input does or does not "correctly describe" the external universe.) Indeed, we don't even require that $Y_q(\hat{u}) = q$. Even if the computer gets confused about what question it's answering, we give it credit if it comes up with the correct answer to our question. In addition, consider some intelligibility function f and associated x. Then recall that we do not even forbid the possibility of two $\hat{u}$'s that are both consistent with that x and that both obey $Y_\alpha(\hat{u}) = f(\hat{u})$, but that nonetheless have different $Y_\alpha(\hat{u})$. ('Accordingly, lack of predictability implies merely that for some f a correct answer cannot be guaranteed, rather than that a wrong answer is assured.

Furthermore, while motivated by the task of predicting the future, the definition of weak predictability presented here is more general, concerning any computation that can be cast in terms of inputs, questions about the universe, and associated answers. For example, no times like 0, τ or T occur in the definition of 'predictable' or in any of the terms going into that definition. Moreover, even when there *is* some temporal ordering that relates the inputs, the outputs, and the prediction involved in the computation, we need not have $T > \tau > 0$ as in Ex. 1. We could just as easily have $T < \tau < 0$ or even $T < 0 < \tau$. So the results presented below will establish the unpredictability *of the past* as well as of the future. They also can be viewed as establishing the fallibility of any observation apparatus and of any control apparatus. These points will be returned to below.

Finally, it is important to realize that the requirement of intelligibility can be removed from the definition of predictability, and many of the results presented below will still hold (e.g., Thm. 2 will still hold). That requirement can be helpful in extensions of this paper's analysis however, and certainly seems "natural". Hence its inclusion in our definition. See the discussion leading up to Def. 4 in the appendix for more discussion of this point.

2. THE UNPREDICTABILITY OF THE FUTURE

i) The impossibility of assuredly correct prediction

Even if we can pose all the questions in some set to a computer, that says nothing about whether by appropriate choice of input that computer can always be assured of correctly answering any question from that set. In fact, even if we restrict attention to question-independent intelligibility sets, no physical computer can be assuredly correct in its predictions concerning the future.

Whereas the impossibility expressed by Thm. 1 follows from cardinality arguments and the power set nature of intelligibility sets, the impossibility of assuredly correct prediction follows from the presence of the negation operator in even (question-independent) intelligibility sets. As an example of the logic underlying the proof, consider a pair of computers predicting the future as in Ex. 1. Have both of the computers have answer subsections that are binary, and have initialization time equal 0 and question time equal T. Have one of the two computers predict the other's time T output bit and then halt and freeze its output, all by some time $\tau < T$, whereas that other computer predicts the negation of the first one's time T output bit just before it too halts. Since both computers' output calculations must halt by τ, they will contradict each other when the prediction time T arrives. Therefore they cannot both be correct in their predictions.

This kind of reasoning can be extended to apply to any pair of physical computers, not just ones that work as in Ex. 1. For example, no "halting and freezing" is required in general. (Indeed, in practice C cannot guarantee that its output will be frozen with a particular output value that does not change until after some time τ, since it is always possible that an outside system comes in and perturbs C.) Even the times 0, τ, and T are superfluous. This is formally stated in the following theorem:

Theorem 2: Consider any pair of distinguishable physical computers $\{C^i : i = 1, 2\}$. It is not possible that both $C^1 > C^2$ and $C^1 < C^2$.

It should be emphasized that Thm. 2 holds no matter how large and powerful our computers are; it even holds if the "physical system underlying" one or both of our computers is the whole universe. It also holds if instead C^2 is the rest of the physical universe external to C^1. As a particular instance of this latter case, the theorem holds even if C^1 and C^2 are physically isolated from each other $\forall\ t > 0$. (Results similar to Thm. 2 that rely on physical coupling between the computers are presented in [33].)

Rather than viewing it as imposing limits on computers, Thm. 2 can instead be viewed as imposing limits on the computational capabilities of the universe as a whole. From this perspective that theorem establishes that the universe cannot support parallel computation in which all the nodes are sufficiently powerful to correctly predict each other's behavior. In addition, it is possible to generalize this paper's formalism to stochastic universes and/or computers. In that extension Thm. 2 takes the form of saying it is impossible for the probability of correct prediction for two computers to both equal 1. An open question is what the highest ε is such that two computers can simultaneously have it as their probability of correct prediction. (See discussion in the appendix just before Lemma 1.)

ii) Implications of Thm. 2

Let C be a computer supposedly capable of correctly predicting the future of any system S if appropriate information concerning the initial state of S is provided to C, as in Ex. 1 above. Assume that C is not so powerful that the universe is incapable of supporting a copy of C in addition to the original. (This is certainly true of any C conceivably built by humans — see the formal definition of a copy of a physical computer in Def. 3 in the appendix.) Have S be such a copy of C. We assume that for any pair of $t = 0$ input values for C, there is at least one world-line of the universe in which C's input is one of those values and the other value constitutes the input of C's copy (i.e., we have input-distinguishability).

Applying Thm. 1 to our two computers, we see that there is a finite intelligibility set that is not intelligible to C, i.e., there are questions concerning an S that cannot even be posed to C. (More formally, there is either such a set for C or for its copy, S.) In addition, by Thm. 2, there is a finite question-independent (and therefore potentially pose-able) intelligibility set concerning S that is not predictable to C. In other words, there must be a question-independent intelligibility function concerning S that C predicts incorrectly, no matter what the input to C.

The binary partition over U_T induced by this unpredictable intelligibility function constitutes a question concerning the time T state of S. In addition every one of the set of potential inputs to C corresponds to a subset of U_0, and therefore corresponds to a subset of the possible states of C's "input section" at time 0. (In Ex. 1, X(.) is set up so that every element in {x} corresponds to one and only one state of C's input section at time 0.) Similarly, every output of C corresponds to a subset of U_τ and therefore a subset of the possible states of C's "output section" at time τ. Accordingly, our result means that there is no input to C at time 0 that will result in C's output at time τ having the correct answer to our question concerning the time T state of S. For $0 < \tau < T$, this constitutes a formal proof that no computer can predict the future faster than it occurs. (Or more precisely, that the universe cannot support more than one copy of such a computer.)

This means, in essence, that Laplace was wrong: even if the universe were a giant clock, he would not have been able to reliably predict the universe's future state before it occurred. Viewed differently, Thm. 2 means that regardless of noise levels and the dimensions and other characteristics of the underlying attractors of the physical dynamics of various, there cannot be a time-series prediction algorithm [9] that is always correct in its prediction of the future state of such systems.

Note that there is no requirement that the initialization time, question time, and/or output time of the computer S's partitions equal 0, T, and τ respectively, the values they have for C. All that is required is that this S be a copy of C. In particular the possibility is allowed that S is a temporal translation of C, either forward or backward in time.

In addition, as mentioned previously, the result also holds when the initialization time is 0 and the output time is some $\tau > 0$, but the question time $T < \tau$. In other words, the computer can run an arbitrarily long time *past* T and still must make mistakes. Perhaps more surprisingly, the result still holds if not only is $T < \tau$, but in addition $T < 0$. In this case the result denies the possibility of assuredly correct "prediction" of what occurred in the time preceding initialization. Intuitively speaking, memory is just as fallible as predicting the future. This should not be surprising. After all, no temporally asymmetric law like the second law arises in our analysis, so all the results *must* be time-symmetric. In fact, the temporally (a)symmetric nature of the laws of the universe are irrelevant to Thm. 2 — that theorem treats the entire universe's world-line as a single entity.

In opposition to this formal proof of the necessary fallibility of retrodiction, one is tempted to argue that no contradiction results if I ask two computers to record each others' past states, only with one of them negated (to try to follow along with the proof of Thm. 2). So the claim that Thm. 2 still holds for $T < 0$ can't be true, it would appear, and infallible retrodiction is allowed. To resolve the conflict between this intuitive argument and the explicitly T-independent nature of the proof of Thm. 2, note that Thm. 2 only says that there is *some* recording at which the computer must fail. The set of all such retrodictions encompasses many that are quite complicated. In particular, the liar's paradox at the heart of Thm. 2 will arise when the recordings concern the dynamic pre-images of those future states that establish the fallibility of prediction the future.

To illustrate this in more detail, first note that if two computers are physically isolated from each other for all time, there is no way each can reliably record the others' past state. So our two putative retrodicting computers must be physically coupled, and therefore must be open systems. Now consider a conventional digital version of such a computer, C, whose output partition elements are labelled by the $t = \tau$ states of its output bits. So each possible output of C is the set of *all possible* states of the entire universe that are consistent with some particular $t = \tau$ pattern of C's output bits. Call such a set, of all possible states consistent with the pattern of C's output bits at time τ, "aligned" with that pattern / time pair. In general, since C is open, a set of states that are aligned with an output pattern of C's at time τ will not dynamically map to a set that is aligned with those bits at an earlier time $T < 0$. (Instead, generically, the temporal projection of those states back in time will be consistent with multiple output patterns over C at that earlier time, with each such pattern accompanied by only a proper subset of all possible associated states of the external universe.) In the language of Ex. 1, while $G'''(U_\tau)$ is defined purely in terms of the $t = \tau$ state of C's output bits, this need not be the case for $G'''(U_{t \neq t})$.

So to induce the liar's paradox we pose to S a question concerning $t = T$ that does not concern some set of states aligned with C's output bits at that time. Rather the question we pose concerns the pre-images (over U) of the individual $t = \tau$ U-space partition elements that index C's $t = \tau$ outputs. The same is true for the computer C's retrodiction concerning S. It is these kinds of questions that establish the fallibility of retrodiction.

While these results concerning both prediction and retrodiction hold if C and S are isolated from one another $\forall$ t > 0, they also hold if C and S are coupled at such times. Indeed, they hold no matter what the form of such coupling. So in particular, we can have the coupling consist of C's "observing" some aspect of S. In fact this is the natural way to try to do retrodiction. Accordingly, the impossibility of unerring retrodiction implies the impossibility of unerring observation.

As a more details example, consider a conventional observation experiment, where what variable in S is observed at time τ is determined by characteristics of the experimental apparatus at that time. In other words, it is determined by certain characteristics of $u(\tau)$, i.e., by certain characteristics of $\hat{u}$, i.e. by where $\hat{u}$ is in a particular partition over $\hat{U}$. Each element in that partition corresponds to a different variable to be observed, i.e., to a different question. So in such conventional observation, there is an implicit question-valued partition of $\hat{U}$. The "observation" consists of providing an answer to some associated question. In other words, in conventional observation the choice of what to observe, together with the resultant observation, constitutes an output partition. The input partition initializing the experiment then is a way of forcing (a $\hat{u}$ which gives) an output partition with the desired question, hopefully also having the correct associated answer. (Note that in this interpretation of a physical computer as an observation device, its input will in general not uniquely fix its output answer, unlike the case with prediction discussed in Ex. 1.)

So observation is simply an instance of physical computation. As a result, Thm. 2 establishes the impossibility of a device C that can, infallibly, take any specification of some characteristic of the universe as input, and then observe the value of that characteristic. This impossibility holds independent of considerations of light-cones and the like, and in fact holds just as well in a universe with $c = \infty$ as it does in ours. (Alternatively, the time at which the characteristic is to be observed can be specified in the computer's input, and therefore can be far enough into the future so that the light-cone emanating from the setting of that input can intersect with that of the characteristic being observed.) In all this, Thm. 2 establishes that any putative general-purpose observation apparatus must, for some system to be observed, make a mistake in its claimed observation of that system.

This unobservability constitutes a sort of non-quantum-mechanical "uncertainly principle".

Just like the Copenhagen version of the quantum mechanics uncertainty principle, the physical computation uncertainty principle relies on having an “intelligent” system perform the observation. In contrast to the quantum mechanics case however, in the physical computation version of the uncertainty principle, an “intelligent observational system” is given a formal definition (as a physical computer). See also the discussion in Sec. 4(i).

Just as the impossibility of unerring prediction of the past means that unerring observation is impossible, so the impossibility of unerring prediction of the future means that unerring control is impossible. More precisely, there is nothing in the math that forces C to play a “passive observational role” in the coupling with S. So we can just as well view Thm. 2 as establishing the impossibility of an apparatus capable of ensuring that there is no discrepancy between a value in its “answer section” and an associated characteristic of a system S external to C. (Note that while weak predictability does not require that x fixes the value of y_α independent of the initial state of S, nor does it forbid x to fix y_α; it only requires that y_α correctly answers the associated question concerning S.) Accordingly, there is no such thing as a general-purpose controller that works perfectly, in all situations.

These impossibility results hold even if one tries to have the input to the computer explicitly contain the correct value of the prediction or observation. (Note that since the universe is single-valued and deterministic, such a value must exist.) Impossibility also obtains if the input is stochastic, since it holds for each input value individually.

3. THE MATHEMATICAL STRUCTURE RELATING PHYSICAL COMPUTERS

There is a rich mathematical structure governing the possible predictability relationships among sets of physical computers, especially if one relaxes the presumption that they are pairwise input-distinguishable. This section presents some of that structure.

i) The graphical structure over a set of computers induced by weak predictability

Thm. 2 directly addresses predictability relations within pairwise-distinguishable sets of multiple computers. However one can also use it to derive results for the predictability relationships within other types of sets of computers. For example, consider a set of n physical computers $\{C^i\}$ such that $C^1 > C^2 > ... > C^n > C^1$. If that set is only pairwise-distinguishable, we can have $C^1 > C^2 > ... > C^n$ but still *not* have $C^1 > C^n$. (See Ex. 2" in the appendix.) So it would seem that Thm. 2 does not preclude having $C^n > C^1$, i.e., does not preclude predictability cycles. It turns out though that such cycles are impossible if one considers sets that are more than just pairwise distinguishable. An example is the following corollary of Thm. 2:

Corollary 2: It is not possible to have a (fully) distinguishable set of n physical computers $\{C^i\}$ such that $C^1 > C^2 > ... > C^n > C^1$.

What are the general conditions under which two computers can be predictable to one another? By Thm. 2, we know they aren't if they're input-distinguishable. What about if they're one and the same? No physical computer is input-distinguishable from itself, so Thm. 2 doesn't apply to this issue. However it still turns out that Thm. 2's implication holds:

Theorem 3: No physical computer is predictable to itself.

Intuitively, this result follows from the fact that a computer cannot make as its prediction the logical inverse of its prediction. An important corollary of this result is that no output partition, considered in isolation of any input partition, is predictable to a physical computer that has that output partition. Combining Thm. 3 and Coroll. 2 and identifying the predictability relationship with an edge in a graph, we see that fully distinguishable sets of physical computers constitute (unions of) directed acyclic graphs. The allowed graphical structure of other kinds of sets (e.g., pairwise-distinguishable ones) is not well-understood at present.

ii) God computers, omniscience, and variants of error correction

When considering sets of more than two computers, it is important to realize that while it is symmetric, the input-distinguishability relation need not be transitive. Accordingly, separate pairwise distinguishable sets of computers may partially "overlap" one another. Similarly, stipulating the values of the inputs of any two computers in a pairwise-distinguishable set may force some of the other computers in that set to have a particular input value.

Coroll. 2 does not apply to a pairwise-distinguishable set. To analyze such sets, first define a **god computer** to be any physical computer in a set of computers such that all other physical computers in that set are predictable to the god computer. By Thm. 2, no pairwise-distinguishable set of computers can contain more than one god computer. There is at most one computer in any pairwise distinguishable set that can correctly predict the future of all other members of that set, and more generally at most one that can accurately predict the past of, observe, and/or control any system in that set.

Even a god computer in a pairwise-distinguishable set may not be able to correctly predict all other computers in its set *simultaneously*. The input value it needs to adopt to correctly predict some C^2 may preclude it from correctly predicting some C^3 and vice-versa. One way to analyze this issue is to consider a composite partition $Y^{2 \times 3}$ defined by the output partitions of C^2 and C^3. We can then investigate whether and when our god computer can weakly predict the composite output partition. To that end, define a computer C^1 in a set of pairwise-distinguishable computers $\{C^1, C^2, ...\}$ to be **omniscient** if the composite output partition $Y^{2 \times 3 \times \cdots}$ is predictable to C^1. It is straight-forward to verify that an omniscient computer is a god computer.

Now in general, one might presume that two non-god computers in a pairwise-distinguishable set could have the property that, while individually they cannot predict everything, considered jointly they would constitute a god computer, if only they could work cooperatively. An example of such cooperativity would be having one of the computers predict when the other one's prediction is wrong. It turns out though that under some circumstances the mere presence of some other third computer in that pairwise distinguishable set may make such error-correction impossible, if that other computer is omniscient.

As an example of this, say we have three pair-wise distinguishable computers C^1, C^2, C^3, where C^3 always answers with a bit (i.e., $\nexists\ {y_q}^3$ such that $A({y_q}^3) \nsubseteq \mathbf{B}$). We want C^2's output to "correct" C^3's predictions, and we also want to have those predictions made by C^3 (potentially) concern C^1. So have C^1 be intelligible to C^3. Then it turns out that due to Thm. 2, if C^1 is omniscient, it is not possible that C^2 always correctly outputs a bit saying whether C^3's answer is the correct response to C^3's question. This is stated formally (and then derived) as **Corollary 3** in the appendix. This result even holds if $Y^{2\times 3}$ is only intelligible to C^1, without necessarily being predictable to it.

Coroll. 3 can be viewed as a restriction on the efficacy of any error correction scheme in the presence of a (distinguishable) omniscient computer. There are other restrictions that hold even in the absence of such a third computer. An example arises if we consider two distinguishable mutually intelligible physical computers C^1 and C^2, where both $A({y^1}_q) \subseteq \mathbf{B}$ and $A({y^2}_q) \subseteq \mathbf{B}\ \forall\ {y^1}_q \in \{{y^1}_q\}$ and ${y^2}_q \in \{{y^2}_q\}$. For such computers, it turns out that Thm. 2 means that it is impossible for C^1 and C^2 to be "anti-predictable" to each other, in the sense that for each of them, the prediction they make concerning the state of the other can always be made to be wrong by appropriate choice of input. This is proven as **Corollary 4** in the appendix.

iii) Physical computation analogues of Turing Machines

There are several ways that one can relate the mathematical structure of physical computation to that of conventional computer science. Here we sketch the salient concepts for one such relation coupling physical computation and the mathematical structure governing Turing machines (TMs).

A TM is a device that takes in an input string on an input tape, then based on it produces a sequence of output strings, either "halting" at some time with a final output string (when an internal "halt" state is entered), or never halting. As an alternative, the fact that the halt state has /

hasn't been entered by any time can be reflected in a special associated pattern in the output string, in which case the sequence of output strings can always be taken to be infinite. As explicated above, in the real world inputs and (sequences of) outputs are elements of partitions of $\hat{U}$. So in one translation of TMs to physical computers, strings on tapes are replaced with elements of the partitions X(.) and Y(.).One way of doing this is to have $\{x\}$ be the set of all strings., $\{y_q\}$ then consists of a single partition q that divides up $\hat{U}$ the exact same way as the input partition does, only with A(q) being the set of all infinite sequences of strings. For any $\hat{u}$, $X(\hat{u})$ is an input string, and $Y(\hat{u})$ is the associated sequence of strings generated by running the TM on that input string. Having $Y(\hat{u})$ specify both the initial string and the ensuring sequence of strings is analogous to the conventional way of implementing reversible computation [2-6].

Rather than through a set of internal states, read/write operations, state-transition rules, etc., the transformation of inputs to outputs in a physical computer is achieved simply through the definition of the pair of an associated input partition and output partition. For such a TM that declares in its output string whether it has halted, the physical computation analogue of whether a computation will ever halt is simply whether $\hat{u}$ is in some special subset of $\{y\}$.

In the real world X(.) and Y(.) usually divide up $\hat{U}$ differently. In this they are analogous to TM's with multiple tapes rather than conventional single-tape TMs. One way to generalize this, motivated by the definition of predictability, is to require of each $q \in \{y_q\}$ that A(q) is the set of all possible sequences of strings. Different $q \in \{y_q\}$ are then interpreted as equivalent to questions "what sequence of output strings ensures from some input string s?" for different s.

In this context the question-independent nature of weak predictability is loosely analogous to a TM's being able to overwrite the "question" originally posed on its tape when producing its "answer" on that tape. We will adopt this identification from now on, identifying the physical computation analogue of a TM as an input partition together with the answer component of an output partition.

This identification motivates several analogues of the Halting theorem. Since whether a particular physical computer C^2 "halts" or not can be translated into whether its output is in a particular region, the question of whether C^2 halts is a particular (question-independent) intelligibility function of C^2. Correctly answering the question of whether C^2 halts means predicting that intelligibility function of C^2. In the context of physical computation it is natural to broaden the issue to concern all intelligibility functions of C^2. Accordingly, in this analogue of the claim resolved for TM's (in the negative) by the Halting theorem, one asks if it is possible to construct a physical computer C^1 that can predict any computer C^2. To answer this, simply consider the case where C^2 is a copy of C^1. By applying Thm.'s 2 and 3 to this case, one sees that the answer is no, in agreement with the Halting theorem. (Even if one strengthens the notion of predictability, as in the next subsection, the answer is still no, by Thm. 6 presented below. See also Coroll. 4 in the appendix.)

There exist a number of alternative physical computer analogues of the Halting problem. Though not pursued at length here, it is worth briefly presenting one such alternative. This alternative is motivated by arguing that, in the real world, one is not interested so much in whether the computation will ever "halt", but rather whether the associated output (say conventionally "read" at some pre-fixed time) is "correct". If we take "correct" to be relative to a particular question, this motivates the following alternative analogue of the Halting theorem: Given any set of physical computer $\{C^i\}$, there is no member of that set C such that for every $C' \in \{C^i\}$, (i) C' is intelligible to C; and (ii) for all questions $q' \in \{y'_q\}$, there is an x value which induces C to answer with a 1 if and only if the answer of C' to q' is correct. See **Theorem 4** in the appendix.

iv) Strong predictability

At the other end of the spectrum from distinguishable computers is the case where one computer's input can fix another's, by being observed by that other computer (or perhaps even by setting that other computer's input more directly). It is when such relationships hold that physical computation analogues of various members of the Chomsky hierarchy, and particularly universal Turing machines, arise.

To capture such a relationship, we say that a computer C^2 is **strongly predictable** to C^1 (or equivalently that C^1 **can strongly predict** C^2), and write $C^1 >> C^2$ (or equivalently $C^2 << C^1$) if two conditions hold. First, C^2 must be intelligible to C^1. Second, for every intelligibility function concerning C^2, f, and for every x^2, $\exists\, x^1 \in \{x^1\}$ that **strongly induces** the pair (f, x^2). That is, there exists a value of x^1 such that $X^1(\hat{u}) = x^1$ forces $Y^1{}_p(\hat{u})$ to equal (A(f), f($\hat{u}$)) and reflects the fact that $X^2(\hat{u}) = x^2$ (or, viewed alternatively, forces it to be the case that $X^2(\hat{u}) = x^2$).

If C^1 can strongly predict C^2, then for any x^2 and associated answer $y^2{}_\alpha$ — for any computation C^2 might undertake — there is an input to C^1 that is uniquely associated with x^2 and that causes C^1 to output (any desired question-independent intelligibility function of) the associated y^2. By also ensuring that $X^2(\hat{u}) = x^2$, with that input we ensure that C^1 is outputting C^2's conclusion for the desired premise, x^2. Intuitively, there is some invertible "translating" map that takes C^2's input and "encodes" it in C^1's input, in such a way that C^1 can "emulate" C^2 running on C^2's input, and thereby produce C^2's associated output. In this way C^1 can emulate C^2, much like universal Turing machines can emulate other Turing machines. (See the definition of a universal physical computer below.)

Strong predictability of a computer implies weak predictability of that computer. (Unlike with weak predictability, there is no such thing as strong predictability of a partition.) So results concerning weak predictability that are not predicated on input distinguishability (which is impossible for strong predictability) still hold if they are changed by replacing weak predictability with strong predictability. This includes in particular Thm. 3 and Coroll. 2 (but not Thm. 2).

Weak predictability does not imply strong predictability however. Moreover, the mathematics for sets of physical computers some of which are strongly predictable to each other (and therefore not distinguishable) differs in some respects from that when all the computers are distinguishable (the usual context for investigations of weak predictability). An example is the following result, which shows that strong predictability always is transitive, unlike weak predictability.

Theorem 5: Consider three physical computers $\{C^1, C^2, C^3\}$, and a partition π, where both C^3 and π are intelligible to C^1.

i) $C^1 >> C^2 > \pi \Rightarrow C^1 > \pi$;

ii) $C^1 >> C^2 >> C^3 \Rightarrow C^1 >> C^3$.

Strong predictability also obeys the following result which is analogous to both Thm.'s 2 and 3:

Theorem 6: Consider any pair of (not necessarily distinguishable) physical computers $\{C^i: i = 1, 2\}$. It is not possible that both $C^1 >> C^2$ and $C^1 << C^2$.

Many of the conditions in the preceding results can be weakened and the associated conclusions still hold (e.g., we can weaken the restriction that intelligibility functions have image space $\subseteq$ **B**.) These weakened version are usually more obscure though, which is why they are not presented here.

A TM T^1 can emulate a TM T^2 if for any input for T^2, T^1 produces the same output as T^2 when given an appropriately modified version of that input. (Typically, the "modification" involves pre-pending an encoding of T^2 to that input.) The analogous concept for a physical computer is strong predictability; one physical computer can "emulate" another (not distinguishable, in general) computer if it can strongly predict that other one. Intuitively, the two components of T^1's emulating T^2, involving T^2's input and its computational behavior, respectively, correspond to the two components of the requirement concerning x^1 values that occur in the definition of strong predictability. The requirement that the x^1 value forces the answer of y^1 to equal that of any intelligibility function of C^2 is analogous to encoding (the computational behavior of) the TM T^2 in a string provided to the emulating TM, T^1. Requiring as well that the value x^1 ensures that $X^2(\hat{u}) = x^2$ is analogous to also including an "appropriately modified" version of T^2's input in the string provided to T^1. (Note that any mapping taking $x^2 \in \{x^2\}$ to an x^1 that in turn induces that starting x^2 is invertible, by construction.) This motivates the following definition of the analogue of a universal TM:

Definition 9: A **universal** physical computer for a set of physical computers is a member of that set that can strongly predict all other members of that set.

Note that rather than reproduce the output of a computer it is strongly predicting, a universal physical computer produces the value of an intelligibility function applied to that output. This allows the computers in our set to have different output spaces from the universal physical computer. However it contrasts with the situation with conventional TM's, being a generalization of such TM's.

v) Prediction complexity

In computer science theory, given a universal TM T, the algorithmic complexity of an output string s is defined as the length of the smallest input string s' that when input to T produces s as output. To construct our physical computation analogue of this, we need to define the "length" of an input region of a physical computer. To do this, first, given any computer C and partition π of $\hat{U}$, define a **(weak) prediction input set** as a minimal subset of x values needed for C to weakly induce all intelligibility functions of π. $C^{-1}(\pi)$ is defined as the set of all such prediction input sets. Intuitively, the prediction set of C for π / C' is a minimal subset of {x} that is needed by C for π / C' to be predictable to C.

Next, to define the physical computation analogue of the length of a string, given a computer C define the **length** of a subset of $\Xi \subseteq \{x\}$ as the negative logarithm of the volume of all $\hat{u} \in \hat{U}$ such that $X(\hat{u}) \in \Xi$. We write this as $\mathbf{l}(\Xi)$. Then if $C > \pi$ (so $C^{-1}(\pi)$ is non-empty), the **prediction complexity** of π for C is the minimal such length over the set $C^{-1}(\pi)$. We write that complexity as $\mathbf{c}(\pi \mid C)$. (Note that the prediction complexity is defined in terms of weak predictability rather than strong; strong predictability will arise in our bounds on it.)

We are primarily interested in prediction complexities of binary partitions, in particular of the binary partitions induced by the separate single elements of multi-element partitions. (The binary partition induced by some particular element $p \in \pi'$ is just the binary-valued function of $\hat{u}$ of whether or not $\pi'(\hat{u}) = p$.) To see what our definitions mean for such a partition, say you are given some set $\sigma \subset \hat{U}$ (i.e., you are given a binary partition of $\hat{U}$). Suppose further that you wish to know whether the universe is in σ, and you have some computer C to use to answer this question (i.e., evaluate all four intelligibility functions of the partition $(\sigma, \hat{U} \setminus \sigma)$. Then loosely speaking, the prediction complexity of σ with respect to C is the minimal amount of Shannon information that must be imposed in C's inputs in order to be assured that C's output correctly answers that question. In particular, if σ corresponds to a potential future state of some system S external to C, then $\mathbf{c}(\sigma \mid C)$ is a measure of how difficult it is for C to predict that future state of S.[3] Loosely speaking, the more sensitively that future state depends on current conditions, the more complex it is.

In many situations it will be most natural to choose the volume measure implicitly defining $\mathbf{l}(.)$ to be uniform over accessible phase space volume, so that the complexity of Ξ is the negative physical entropy of constraining $\hat{u}$ to lie in Ξ. But that need not be the case. For example, we can instead define the measure so that the volume of each element of the associated $\{x\}$ is some arbitrary positive real number. In this case, the lengths of the elements of $\{x\}$ provides us with an arbitrary ordering over those elements.

The following example illustrates the connection between lengths of regions Ξ and lengths of strings in TM's:

Example 3: In a conventional computer (see Ex. 1 above), we can define a "partial string" s (sometimes called a "file") taking up the beginning of an input section of memory as the set of all "complete strings" taking up the entire input section whose beginning is s. We can then identify the input to the computer as such a partial string in its input section. (Typically, there would be a special fixed-size "length of partial string" region even earlier, at the very beginning of the input section, telling the computer how much of the complete string to read to get that partial string.) If we append certain bits to s to get a new longer input partial string, s', the set of complete strings consistent with s' is a proper subset of the set of complete strings consistent with s. Assuming our volume measure $d\mu$ is independent of the contents of the "length of partial string" region, this means that $\mathbf{l}(s') \ge \mathbf{l}(s)$.

This is in accord with the usual definition of the length of a string used in Turing machine theory. Indeed, if s' contains n more bits than does s, then there are 2^n times as many complete strings consistent with s as there are consistent with s'. Accordingly, if we take logarithms to have base 2, $\mathbf{l}(s') = \mathbf{l}(s) + n$.

Say we want our computer to be able to predict whether $\hat{u}$ lies in some set σ. (To maintain the analogy with Turing machines, σ could delineate an "output partial string". This could be done for example by delineating a particular value of a prediction, perhaps even one in some other computer.) In the usual way, this corresponds to having the binary partition $\{ \hat{u} \in \sigma, \hat{u} \notin \sigma\}$ be weakly predictable to our computer. So the prediction complexity of that prediction is the length of the shortest region of our input space that will weakly induce that prediction. (Note that since we require that all four intelligibility functions of σ be induced, more than one input "partial string" is required for that induction, in general.)

We now derive a bound on differences of the prediction complexity of a partition with respect to two different universal computers. First, given C together with some other computer C', we need to define a **strong prediction input set** of C for the triple of (C', a subset Ξ' of the input values of C', and a subset $f\,'$ of the intelligibility functions for C'). This is a minimal subset of C's input values needed to strongly induce every pair $(f' \in f\,', x' \in \Xi')$. When there is at least one such subset we will write $C^{-1}(C', \Xi', f\,')$ for the set of all such subsets.

The fact that y_p values (cf. the definition of prediction partition in the appendix) specify the set $A(y_q)$ makes working with Def.'s 10 and 11 a bit messy. In particular, to relate prediction complexity to properties of the associated universal physical computer we must use a set of "identity" intelligibility functions defined as follows:

Definition 12 (i): Given a space $Z \subseteq \mathbf{B}$ and a physical computer $C = (X, Y)$,

$\{I^C{}_Z\}$ is the set of all question-independent intelligibility functions of C where $A(I^C{}_Z) = Z$, and where $\forall\ \hat{u}$ such that $A(Y_q(\hat{u})) = Z$, $I^C{}_Z(\hat{u}) = Y_\alpha(\hat{u})$.

We also will need the following definition:

Definition 12 (ii): Given a space $Z \subseteq \mathbf{B}$ and a physical computer C = (X, Y), "$C^{\leftarrow}(Z)$" is defined as those $x \in \{x\}$ such that $X(\hat{u}) = x \Rightarrow A(Y_q(\hat{u})) \subseteq Z$.

So for example, if $Z = \mathbf{B}$, a pair $(x^2 \in [C^2]^{\leftarrow}(Z), I^2{}_Z \in \{I^2{}_Z\})$ is an input to C^2 and an intelligibility function of C^2's output, respectively. That input x^2 induces an associated output question, $q^2 \in \{y^2{}_q\}$, that takes on (both) **B** values as one varies over the $\hat{u}$ input to it. Similarly, the intelligibility function $x^2{}_Z$ takes on (both) **B** values as one varies over the inputs to it.

Using these definitions, we now bound how much more complex a partition can appear to C^1 than to C^2 if C^1 can strongly predict C^2. Though somewhat forbidding in appearance, intuitively, the bound simply reflects the complexity cost of "encoding" C^2 in C^1's input.

Theorem 7: Given any partition π and physical computers C^1 and C^2 where $C^1 >> C^2 > \pi$,

i) $\mathbf{c}(\pi \mid C^1) - \mathbf{c}(\pi \mid C^2) \le$

$$\ln[o(2^\pi)] - \ln[3] + \max_{\{Z \subseteq \mathbf{B},\ x^2 \in [C^2]^{\leftarrow}(Z),\ I^2{}_Z \in \{I^2{}_Z\}\}} \mathbf{l}[\,(C^1)^{-1}(C^2, x^2, I^2{}_Z)\,] - \min_{\{Z \subseteq \mathbf{B},\ x^2 \in [C^2]^{\leftarrow}(Z)\}} \mathbf{l}[\,x^2\,]\,,$$

or alternatively,

ii) $\mathbf{c}(\pi \mid C^1) - \mathbf{c}(\pi \mid C^2) \le$

$$\ln[o(2^\pi)] + \min_{\{Z \subseteq \mathbf{B},\ x^2 \in [C^2]^{\leftarrow}(Z),\ I^2{}_Z \in \{I^2{}_Z\}\}} \mathbf{l}[\,(C^1)^{-1}(C^2, x^2, I^2{}_Z)\,] - \min_{\{Z \subseteq \mathbf{B},\ x^2 \in [C^2]^{\leftarrow}(Z)\}} \mathbf{l}[\,x^2\,]\,.$$

As one varies π, in both bounds in Thm. 7 how the bound depends on C^1 and C^2 does not change. In addition, those bounds are independent of π for all π sharing the same cardinality. So in particular they are independent of the precise choice of partition π so long as it is a binary partition like those discussed in Ex. 3. In addition, intuitively speaking, the term $\mathbf{I}[\, (C^1)^{-1}(C^2, x^2, \;\; I^2{}_Z)$ $]$ occurring in both bounds is related to the cost of emulating the one computer on the other. This illustrates how Thm. 7 is the physical computation analogue of the result in Turing machine theory that the difference in algorithmic complexity of a fixed string with respect to two separate Turing machines is bounded by the complexity of "emulating" the one Turing machine on the other, independent of the fixed string in question.

Consider the possibility that for the laws of physics in our universe, there exist partitions X(.) and Y(.) that constitute a universal physical computer C^* for all other physical computers that exist in our universe. Then by Thm. 6, no other computer is similarly universal. Therefore there exists a unique prediction complexity measure that is applicable to all physical computers in our universe, namely complexity with respect to C^*. (This contrasts with the case of algorithmic information complexity, where there is an arbitrariness in the choice of the universal TM used.) If instead there is no universal physical computer in our universe, then every physical computer C must fail at least once at (strongly) predicting some other physical computer. (Note that unlike the case with weak predictability considered in Thm. 2, here we aren't requiring that the universe be capable of having two distinguishable versions of C.) This establishes the following:

Theorem 8: Either there cannot be computer that strongly predicts all others that exist in our universe, or there is a unique universally applicable complexity measure in our universe.

Similar conclusions hold if one restricts attention to a set of (physically localized) conventional physical computers (cf. Ex. 1), where the light cones in the set are arranged to allow the requisite information to reach the putative universal physical computer. See also the discussion of realities below.

4. DISCUSSION

i) In what sense might reality "be" a computer?

None of the analysis in this paper requires that the possible states of the universe all be characterizable by a single set of often-repeated very regular patterns encapsulated in some concise "physical laws". The results still hold if each $\hat{u} \in \hat{U}$ is just an arbitrary temporally-indexed collection of events, with little to no discernible regularity relating those events. Broadening the interpretation further, whereas in a deterministic universe u(t) uniquely sets all u(t' ≠ t), nothing in our analysis relies on having that or any other kind of structure apply to each $\hat{u}$. Determinism itself is not needed. In fact, $\hat{U}$ can be any kind of set whatsoever, even one whose individual elements cannot reasonably be viewed as "time-indexed collections of events", regular or otherwise, or even one whose elements are not vectors, and our results still hold.

As mentioned in the introduction, several authors have speculated that the entire universe and its physical laws are not some underlying structure *governed* by the conclusions of a computer, but rather in some sense *are* a computer, without any extraneous "underlying structure". In light of the breadth of the possible $\hat{U}$ to which this paper's analysis applies, it is interesting to consider this issue when "computer" is interpreted to mean a physical computer. This use of the mathematics of physical computation implicitly differs from the analysis up to now in which $\hat{u}$ is a time-ordered collection of events that contains a computer, embodied in some subset of its degrees of freedom. In contrast, here $\hat{u}$ can be completely arbitrary, and our partitions are allowed to involve all of the degrees of freedom of $\hat{u}$, not just some subset of them. More importantly, while we still identify a particular instantiation of the laws of the universe with a $\hat{u}$, we do not identify what are intuitively viewed as the "physical properties" of that instantiation directly with that $\hat{u}$, per se. Rather we collectively identify all of those physical properties — the totality of what is observable to humans concerning the universe — as the triple of (a computational answer, to a particular (high-dimensional) question, in response to a particular input). The precise such triple is the one that is induced by that $\hat{u}$ in concert with the X(.) and Y(.) of some physical computer. So here a particular $\hat{u} \in \hat{U}$, by itself, has no "physical meaning" whatsoever; it is the input-question-answer triple that it induces, via X(.) and Y(.), that provides all such meaning. Without that associated triple, $\hat{u}$ is just a point in a set, with no ancillary structure that could imbue it with meaning.

This identification of all physically meaningful properties of the universe $\hat{u}$ with the single associated input-question-answer triple of a computer has some quite reasonable qualities. For example, the value of the input gives a restriction on $\hat{u}$. Intuitively this restriction is akin to a specification of the physical boundary conditions under which the answer to the question of the universe is calculated. Note that that input value not only fixes the universe's answer though, but also the very question being answered. So for major enough changes to the input, in general there has to be a change in that question being answered by the universe. This too is reasonable; intuitively, the original question is no longer meaningful given a large enough change to $\hat{u}$.

Under this identification, the full mapping from arbitrary inputs to the associated question / answer pairs provides all possible pairs of boundary conditions and associated physical properties of the universe's entire worldline. In other words, that mapping — the computer C — constitutes the laws of the universe. So under this identification we do not need elaborate considerations of grammars, formulations of logic, the foundations of mathematical reasoning etc. to express the laws of the universe. Indeed, since we express the laws via a structure itself defined in terms of $\hat{U}$ (namely C), the states $\hat{U}$ and the laws governing them form a self-contained unit.

To formalize this, we say that a pair ($\hat{U}$, C) is a **reality**. One reality is a **copy** of another if their computers are copies of each other. If two realities are copies, then their law-providing computers have identical relationships between their inputs, the questions they associate with those inputs, and the answers they provide to those questions. Accordingly, we identify a particular set of "laws of the universe" with an equivalence class of realities that are copies of one another, even if the spaces $\hat{U}$ of those realities differ.

Say we are given a reality ($\hat{U}$, C). We can calculate for what sets $\{C^i\}$ of (perhaps non-distinguishable) computers defined over $\hat{U}$ the joint output partition $Y^{1\times 2\times \cdots}$ is predictable to C. Label that set of sets χ. For any $\hat{u}$, C's answer is the value of (an associated intelligibility function of) the outputs of any one of those $\{C^i\} \in \chi$ taken all at once. Next, given some $\hat{u} \in \hat{U}$, there is some subset $\chi(\hat{u}) \subseteq \chi$ of $\{C^i\}$ that are weakly induced by C's associated input, X($\hat{u}$). These, intuitively, are the $\{C^i\}$ that are both predictable to C and are actually predicted by C for the $\hat{u}$ at hand. In a certain sense, if C is the "laws" of the reality, then having $Y^{1 \times 2 \times \cdots}$ for a particular $\{C^i\}$ $\in \chi$ be predictable to C is a minimal condition for saying that the computers in $\{C^i\}$ are "allowed by" or "consistent with" ($\hat{U}$, C). Having X($\hat{u}$) induce that $\{C^i\}$ for the $\hat{u}$ at hand is then a minimal condition for saying that the $\{C^i\}$ are "real", and "exist" in that $\hat{u}$ (cf. Thm. 8). (It is interesting to speculate on the similarity between having multiple sets $\{C^i\} \in \chi(\hat{u})$ and the many worlds interpretation of quantum mechanics.)

Note that by definition of predictability, whether some $\{C^i\}$ "exists" is a function of whether C can correctly give its answers *for* $\hat{u}$ *other than the single one at hand*. This reliance on counterfactual $\hat{u}$ to ascribe existence to a $\{C^i\}$ reflects the fact that a single $\hat{u}$, by itself, contains no information. Even if $\hat{u}$ is a collection of high-dimensional real numbers (e.g., a collection of phase space positions), it has no meaning except in comparison to other such collections.

As a particular example of all this, we can have the elements of $\{C^i\}$ be the entire sequence of predictions/observations that constitute the mind of some particular scientist. Doing so, we see that a reality induces a set of scientists, each given by a different $\{C^i\}$. As another example, the human endeavour comprising the field of physics constitutes a computer, with its input and output partitions delineated by states of the mind(s) of one or more physicists. The goal of the field is to have the computer comprised of those two partitions be computationally equivalent to (i.e., a copy of) that of the embedding reality. The analysis of this paper provides some results concerning the possible relationships between the field of physics and those laws governing our embedding reality. For example, by Thm. 2, if we presume that the minds of physicists are predictable to the laws of the universe, then those laws are not predictable to physicists.

In addition to results concerning human endeavours, the analysis of this paper also provides results concerning sets of mathematical laws governing universes. For example, for finite o($\hat{U}$), it is often reasonable to have one x value for each $\hat{u}$, and similarly one y value for each $\hat{u}$ (that is the maximum possible number of both x's and y's). Since there are $2^{o(\hat{U})}$ binary-valued questions concerning $\hat{U}$, this means the (usually vast) majority of questions are not in $\{y_q\}$. So the "laws of the universe" cannot pose most questions concerning that universe (cf. Thm. 1). Furthermore, by Thm. 3, we know that there are questions q (potentially not in $\{y_q\}$) for which there is no x value that can ensure that C's answer correctly gives q($\hat{u}$). There are questions concerning the universe that we can never force the laws of the universe to answer correctly.

These results are particularly suggestive if we recall that observation is a form of physical computation. In ability to pose all questions therefore implies a "coarse-graining" over the set of possible observations. It is tempting to try to relate this to the quantum mechanical uncertainty principle. Note that this physical computation "uncertainty principle" is different from the Thm. 2-based one discussed in the text. Note also that whereas a particular $\hat{u}$ indices a unique answer, a particular x value and y_q do not. This is suggestive of the indeterminacy of observation in quantum-mechanics — knowing the boundary conditions of the universe and the observational question being posed to it need not uniquely fix the associated answer.

There are a number of stronger variants of all of this that are worth investigating. In particular, one could add other conditions to the definition of whether $\{C^i\}$ "exists". An example would be to incorporate the notion of C strongly inducing intelligibility functions of the $\{C^i\}$. Among other things, this would allow us to define the complexity, to the computer constituting the very laws of the universe, of answering a particular question.

Another example arises in response to the argument that rather than conveying physical meaning, the partitions X and Y are ultimately just arbitrary "interpretations" of $\hat{u}$, with no further physical significance. According to this argument, any other interpretation, any other computer defined over the set of possible $\hat{u}$, can be viewed as just as legitimate. When (as in previous sections of this paper) an electronic workstation constitutes $\hat{u}$, this arbitrariness isn't a problem. It is reasonable to say that the user of the workstation provides the interpretation of $\hat{u}$; it is (s)he who ultimately deems what the inputs and outputs to that workstation "mean". A different user of the exact same workstation undergoing the exact same dynamics is free to interpret that workstations's inputs nd outputs differently, and thereby constitute a different computer C. One might want less freedom of interpretation though if rather than a workstation embedded in a universe and accompanied by an interpreting user in that universe, the computer under consideration is supposed to be the very laws of that universe themselves. This issue can be especially nettlesome when we want to view those laws as unique somehow, independent of any interpreting "user".

This objection is ultimately philosophical, amounting to a semantic disagreement over how to define whether two realities are "the same", i.e., of how to define whether they have the same "laws of the universe". The view expounded above is in favor of a "weak" definition, and simply says that a reality's laws are *not* embodied in $\hat{u}$, but rather in C. An alternative "strong" definition, overcoming the objections raised above, adds conditions to the weak definition. These conditions inextricably couple $\hat{u}$ and C, via the relationship between the question / answer pairs in C and the associated elements in $\hat{U}$, and define two realities to "be the same" if they share that property. Formally, we say that ($\hat{U}$, C) is **computationally equivalent** to a different reality ($\hat{U}'$, C') if two conditions hold. First, the two realities must be copies of each other, so that their computers share the same set-valued function from inputs x to outputs (y_q, y_α) (as in the weak definition). Second, the two computers must share the same set-valued function from inputs to the reality's response to the associated question, i.e., the same function from the value of X($\hat{u}$) to the value of α = $[Y_q(\hat{u})](\hat{u})$.[4] (Note that use of this stronger definition in non way negates the properties involving sets χ expounded at the beginning of this subsection.)

A related way of responding to the objection is to consider realities that do not contradict themselves, i.e., whose computers are infallible (see the discussion in the appendix just before Coroll. 1). Requiring that a physical computer be infallible if we are to identify it as a universe certainly seems reasonable. Moreover, if the computers in two realities are both infallible, then they are copies of each other if and only if they are computationally equivalent. So if we restrict attention to infallible computers, the issue of computational equivalence between realities is reduced to the original issue of whether the realities are copies, and there is no difference between the weak and strong definitions of whether two realities are "the same". In addition, if C is stable (see the appendix), then the issue of whether C weakly predicts some C' simplifies to whether C' is intelligible to C. Note also that for the computers in infallible realities, we can simplify the definition of Y to be just a mapping from $\hat{u}$ to questions (the associated answers being set automatically). For all these reasons, when trying to capture the human concept of what it means for two universes to "be the same", it seems reasonable to concentrate on equivalence classes of infallible realities that are copies of one another.

ii) Relation of Thm. 2 to previous work

Any results concerning physical computation should, at a minimum, apply to the computer lying on a scientist's desk. However that computer is governed by the mathematics of deterministic finite automata, not that of Turing machines. In particular, the impossibility results concerning Turing machines rely on infinite structures that do not exist in any computer on a scientist's desk.

On the other hand, when one carefully analyzes actual computers that perform calculations concerning the physical world, one uncovers a mathematical structure governing those computers that is replete with its own impossibility results. While much of that structure parallels Turing machine theory, much of it has no direct analogue in that theory. For example, it has no need for structures like tapes, moveable heads, internal states, read/write capabilities, and the like, none of which have any obvious relation to the laws of quantum mechanics and general relativity.

Nonetheless, there are a number of previous results in the literature that can be viewed as Turing machine analogues of Thm. 2. Many authors have shown how to construct Turing Machines out of physical systems (see for example [11, 25] and references therein). By the usual uncomputability results, there are properties of such systems that cannot be calculated on a physical Turing machine within a fixed allotment of time (assuming each step in the calculation takes a fixed non-infinitesimal time). In addition, there have been a number of results explicitly showing how to construct physical systems whose future state is non-computable, without going through the intermediate step of establishing computational universality [14, 26].

There are several important respects in which the results of this paper extend this previous work. All of these previous results rely on infinities of some sort in physically unrealizable systems (e.g., in [26] an infinite number of steps are needed to construct the physical system whose future state is not computable). In addition, they all assume one's computing device is no more powerful than a Turing machine. Also none of them are motivated by scenarios where the computation is supposed to be a prediction of the future. Nor are they extendable to allow arbitrary coupling between the computer and the external universe, as (for example) in the processes of observation and control. There are other limitations that apply to many of these previous results individually, while not applying to each and every one of them. For example, in [26] it is crucial that we are computing an infinite precision real number rather than a "finite precision" quantity like an integer. As another example, many of these previous results explicitly require chaotic dynamics (e.g., [8]). None of these limitations apply to the result of this paper.

iii) Future work

Future work includes investigating the following issues:

i) How are the results modified if one is concerned with probabilities of erroneous prediction rather than just worst-case analysis of whether there can possibly be erroneous prediction?

ii) How must the definitions and associated results be modified for analog computers (so that one is concerned with amounts of error rather than whether there is an error)? Even if one is predicting the future state of a stochastic system, so long as that prediction is falsifiable the analysis in this paper applies. (See the discussion just before Ex. 1.) However how should the analysis be changed if what one is trying to predict is a random variable? Alternatively, what if (as in the classical real world) $\hat{u}$ has a definite value, but the output of the computer is a probability distribution? A preliminary analysis of this is presented just before Lemma 1 in the appendix. There it is proven that there cannot be two computers both of which have a "degree of weak predictability" (a measure quantifying the accuracy of probability distribution output) equal to 1. Just what the upper bound on such a pair of degrees of weak predictability is unknown.

iii) Since by adopting the many-world interpretation we can cast quantum mechanics as purely deterministic evolution in Hilbert space, the presumption of determinism in this paper does not *a priori* invalidate its applicability to quantum systems. However it is still worth asking whether there any modifications to the definitions that would facilitate the analysis for quantum systems, especially if we adopt the Copenhagen interpretation. If there are such modifications, then how are the ensuing results different for quantum systems? (As an example of such a modification, one might want to allow sufficient time between T and τ to not run into difficulties due to the Heisenberg uncertainty principle.)

iv) Find the exact point of failure — which according to (1) and (2) must exist — of the intuitive argument "If the computer is simply a sufficiently large and fast Hamiltonian evolution approximator, then it can emulate any finite classical non-chaotic system".

v) As mentioned in the introduction, there is a large body of work showing how to embed TM's in physical systems. One topic for future work is following an analogous program in the domain of physical computation, for example by investigating what physical systems support copies of any element of various sets of physical computers.

vi) Exploiting the generality of our definitions, it may be possible to apply the analysis of this paper to the foundations of mathematics. As an example, view each $\hat{u} \in \hat{U}$ as a "book". Each book consists of a collection of mathematical propositions, for example (though not necessarily) expressed as strings over some fixed alphabet. The precise choice of $\hat{U}$ can embody any desired restrictions on the set of possible books. The pair of a question and answer then is a choice of a subset of books in $\hat{U}$. For example, such a pair could be a subset of books all of which contain propositions that all "make the same claim" (i.e., give the same answer) concerning some formal mathematical hypothesis (i.e., concerning the question at hand). Next, a choice of an input to a computer is a restriction of attention to a certain set of books. So as an, it could be a restriction to a set of books all of which adhere to a certain set of axioms (that set constitute the premise that is input to the computer). Finally, the output function is a mapping from a book to a question and answer. For example, $\hat{U}$ may be *a priori* restricted to books that contain declarations of the sort "given these axioms, the following is true". In that case, the output function is a way of choosing a single such declaration from each book. (By allowing only one question per book, the output function manages to sidestep the issue of ensuring no contradiction arises between its answers to various questions for the same underlying book.)

vii) What other restrictions are there on the predictability relations within distinguishable sets of physical computers beyond that they form unions of DAG's? In other words, which unions of DAG's can be manifested as the predictability relations within a distinguishable set? How does this answer change depending on whether we are considering sets of fully input-distinguishable computers or sets of pairwise-distinguishable computers? For what computers are there finite / countably infinite / uncountably infinite numbers of levels below it in the DAG to which it belongs? Might such levels be gainfully compared to the conventional computer science theory issue of position in the Chomsky hierarchy?

viii) One might try to characterize the unpredictability-of-the-future result of Thm. 2 is as the physical computation analogue of the following issue in Turing machine theory: Can one construct a Turing machine M that can take as input A, an encoding of a Turing machine and its tape, and for any such A compute what state A's Turing machine will be in after will be in after n steps, and perform this computation in fewer than n steps? This characterization suggests investigating the formal parallels (if any) between the results of these papers and the "speed-up" theorems of computer science.

ix) More speculatively, the close formal connection between the results of this second paper and those of computer science theory suggest that it may be possible to find physical analogues of most of the other results of computer science theory, and thereby construct a full-blown "physical computer science theory". In particular, it may be possible to build a hierarchy of physical computing power, in analogy to the Chomsky hierarchy. In this way we could translate computer science theory into physics, and thereby render it physically meaningful.

We might be able to do at least some of this even without relying on the DAG relationship among the physical computers in a particular set. As an example, we could consider a system that *can* correctly predict the future state of the universe from any current state of the universe, before that future state occurs. The behavior of such a system is perfectly well-defined, since the laws of physics are fully deterministic (for quantum mechanics this statement implicitly presumes that one views those laws as regarding the evolution of the wave function rather than of observables determined by non-unitary transformations of that wave function). Nonetheless, by the central unpredictability result of Thm. 2, we know that such a system lies too high in the hierarchy to exist in more than one copy in our physical universe.

With such a system identified with an oracle of computer science theory we have the definition of a "physical" oracle. Can we construct further analogues with computer science theory by leveraging that definition of a physical oracle? In other words, can we take the relationships between (computer science) oracles, Turing machines, and the other members of the (computer science) Chomsky hierarchy, and use those relationships together with our (physical) oracle and physical computers to gainfully define other members of a (physical) Chomsky hierarchy?

x) Can we then go further and define physical analogues of concepts like P vs. NP, and the like? Might the halting probability constant Ω of algorithmic information theory have an analogue in physical computation theory?

As another example of possible links between conventional computer science theory and that of physical computers, is there a physical computer analogue of Berry's paradox? Weakly predicting a partition is the physical computation analogue of "generating a symbol sequence" in algorithmic information complexity. The core of Berry's paradox is that there are numbers k such that no Turing machine can generate a sequence having algorithmic information complexity k (with respect to some pre-specified universal Turing machine U). So for example one closely related issue in physical computation is to characterize the physical computers C^1 and $x \in \Re$ such that $\exists$ a computer C^2 where $C^1 >> C^2$ and where $\forall$ partitions π, C^2 weakly predicts whether $\mathbf{c}(\pi \mid C^1) >$ x (i.e., such that $\exists\ x^2 \in \{x^2\}$ such that $X^2(\hat{u}) = x^2 \Rightarrow Y^2{}_p(\hat{u}) = (\mathbf{B}$, whether $\mathbf{c}(\pi \mid C^1) > x))$.

xi) Concerns of computer science theory, and in particular of the theory of Turing machines, have recently been incorporated into a good deal of work on the foundations of physics (e.g., [36, 37}). Future work involves replacing physical computers for Turing machines in this work, along with replacing notions like prediction complexity for notions like algorithmic complexity.

xii) More generally, there have been many candidates proposed for how one should measure “the complexity” of a physical system, e.g., thermodynamic depth [21], logical depth [5], and physical complexity [36, 37]. Future work involves limning the relation between these alternatives and prediction complexity. Particularly intriguing in this regard is logical depth, which is explicitly concerned with “how much mathematical work” is needed to perform a computation, measured in number of computation steps. Prediction complexity is also concerned with such work, only measured spatially in terms of how much initialization precision is required to perform the computation.

xiii) Other future work involves investigating other possible definitions of complexity for physical computation. Even sticking to analogues of algorithmic information complexity, these might extend significantly beyond the modifications to the definition of prediction complexity discussed in the text. For example, one might try to define the analogue of a bit sequence’s “length” in terms of the number of elements in $\{y_q\}$, rather than in terms of a volume. As another alternative one might take the (inverse) complexity of a computational device to be the number of input-distinguishable computers that can predict that device (working in some pre-specified input-distinguishable set, presumably).

xiv) There are at least several ways that the formal definition of a reality presented in Sec. 4(i)can be modified. For example, one could consider realities that consist of sets of multiple computers together with an underlying universe, rather than just a single such computer. This would bring all the multiple computer unpredictability results (e.g., Thm. 2) directly into play within the fundamental laws of physics themselves. (A number of other topics related to realities that are worth investigating are presented in Sec. 4(i).)

xvi) Originally we restricted attention to intelligibility functions that are question-independent because otherwise no pair of computers could be mutually intelligible (Thm. 1). However it turned out that even with this restriction, no pair of computers can be mutually predictable (Thm. 2). Accordingly, in Sections 3 and 4 attention shifted to god computers, which can correctly predict any computer outside of themselves, but are not themselves predictable to such computers. Given this shift though, Thm. 1 now does not provide a reason to require that our intelligibility functions be question-independent. Future work involves re-analyzing the issues addressed in Sections 3 and 4 for full question-*dependent* intelligibility functions. Other future work involves re-analyzing those issues for changes in which of the conditions (i), (ii) and/or (iii) discussed in the appendix are used to define weak and/or strong predictability.

FOOTNOTES

[1] To "remember", in the present, an event from the past, formally means "predicting" that event accurately (i.e., retrodicting the event), using only information from the present. Such retrodiction relies crucially on the second law. Hence, the temporal asymmetry of the second law causes the temporal asymmetry of memory (we remember the past, not the future). That asymmetry of memory in turn causes the temporal asymmetry of the psychological arrow of time. "Memory systems theory" refers to the associated physics of retrodiction; it is the thermodynamic analysis of systems for transferring information from the past to the present. See [31].

[2] More prosaically, to motivate intelligibility we can simply note that we wish to be able to pose to a computer C^1 any prediction question we can formulate. In particular, this means we wish to be able to pose to C^1 any questions concerning well-defined aspects of the future state of C^2. Now

consider having C^2 be a conventional computer based on an underlying physical system. Then we want to be able to predict C^2's output at time τ as $Y^2(u_\tau)$. Therefore in addition to any other questions we might want to be able to pose to it, we want to be able to pose to C^1 questions involving the value $Y^2(u_\tau)$ (e.g., is that value equal to some x_1? to some x_1 or x_2? to that x_1 or some other x_3? etc.). We want C^1 to "understand" y^2 sufficiently well to be able to pose binary-valued questions concerning it. This is equivalent to requiring intelligibility.

[3] Especially for non-binary π, many other definitions of prediction complexity besides Def. 11(ii) can be motivated. For example, one could reasonably define the complexity of π to be the sum of the complexities of each binary partition induced by an element of π, i.e., one could define it as $\Sigma_{p \in \pi}\, \mathbf{c}(\{\hat{u} \in p,\ \hat{u} \notin p\} \mid C)$. Another variant, one that would differ from the one considered in the text even for binary partitions, is $\min_{\rho \in C^{-1}(\pi)} [\Sigma x_{\in \rho}\, \mathbf{l}(x)]$. For reasons of space, no such alternatives will be considered in this paper.

[4] Note that there is a lot of structure not captured in this definition. As an example, two realities can be computationally equivalent even if they differ in their functions mapping $X(\hat{u}) \rightarrow [Y_q(\hat{u})](X^{-1}(x_1))$, where x_1 is the first element of $\{x\}$ (so that for neither computer does $X^{-1}(x_1)$ vary as the $\hat{u}$ argument to $[Y_q(.)]$ is varied). Such a difference between the two realities is akin to a difference in their responses to counter-factual questions.

APPENDIX: FORMAL DEFINITIONS AND PROOFS

This appendix presents the fully formal definitions and proofs of the results discussed in the text. We start with the following definition:

Definition 1:

i) A (**computation**) **partition** is a pair, consisting of a non-empty set of partition-element labels and a single-valued mapping from $\hat{U}$ into that set. Unless stated otherwise, the mapping is assumed to be surjective onto the set.

ii) Any **question** $q \in Q$ is a partition, whose set of partition-element labels is A, with elements $\alpha \in A$ called **answers** to that question. A(q) indicates the A-component of the pair q. We restrict attention to Q such $\exists$ at least two elements in A(q) for at least one $q \in Q$.

Note that we make no assumptions concerning the finiteness of Q and/or any of the $\{A(q \in Q\}$. Unless indicated otherwise (e.g., in the definition of questions), any partition is assumed to contain at least two elements. Note that the definition of a computer partition differs from that of a conventional set-theoretic partition in its inclusion of the partition-element labels.

Given these definitions, we can now define physical computers:

Definition 2: i) In an **output partition** Y, the space of partition element labels is a space of possible "outputs", $\{y\}$, consisting of all pairs $\{y_q \in Q, y_\alpha \in A(y_q)\}$, for some Q and associated A(.) as defined in Def. (1). Often, for convenience, we will write an output partition Y explicitly in the form (Q, Y), where Y(.) is the output map $\hat{u} \in \hat{U} \rightarrow \{y_q \in Q, y_\alpha \in A(y_q)\}$. Also, we will find it useful to define an associated (**p**rediction) partition, $Y_p(.) : \hat{u} \rightarrow (A(Y_q(\hat{u}), Y_\alpha(\hat{u}))$.

ii) In an **input partition**, X, the space of partition element labels is a space of possible "inputs", $\{x\} \equiv A(X)$.

iii) A (**physical**) **computer** consists of the double of an input partition and an output partition

Since we are restricting attention to non-empty Q (cf. Def. 1), $\{y\}$ is non-empty. The surjectivity usually assumed of X(.) and Y(.) (cf. Def. 1) is a restriction on $\{x\}$ and $\{y\}$, respectively. In the case of Y it reflects the fact that we want the computer to be able to provide any of the allowed answers to any question it can pose. (This property is perhaps the most important reason why we don't define the output of a computer simply to be a region of $\hat{U}$, but rather to be a question-answer pair that delineates such a region.) More generally, for both inputs and outputs, for reasons of convenience we don't want to allow a value "officially" to be in the space of the computer's potential inputs (outputs) if there is no state of the computer that corresponds to that input (output). For example, if the computer is a digital workstation with a kilobyte of its RAM set aside as input, it makes no sense to have the input space contain more than $(2^8)^{1024}$ values, the number of possible bit patterns in that RAM. For an example of when Y(.) need not be surjective, see Def. 7 below.

Example 1 continued: Restrict attention to computers (X, Q, Y) where all $q \in Q$ concern the same moment T. Then you get a different physical computer if you change any of the times 0, T, or τ (implicitly setting X(.(), all $q(.) \in Q$, and Y(.), respectively). In this sense the electronic workstation on your desk is actually a set of many different computers. All those computers are (typically) copies of one another however. This difference with common vernacular is important to bear in mind in considering the results presented below.

We can now define a "copy" of a physical computer:

Definition 2 (iv): Given a computer $C \equiv \{X, Q, Y(.)\}$, define the **implication** in {y} of any value $x \in \{x\}$ to be the set of all $y \in \{y\}$ consistent with x, in that $\exists\ \hat{u} \in \hat{U}$ for which both $X(\hat{u}) = x$ and $Y(\hat{u}) = y$.

v) The computer $C^2 \equiv \{X^2, Q^2, Y^2(.)\}$ is a **copy** of the computer $C^1 \equiv \{X^1, Q^1, Y^1(.)\}$ iff $Q^2 = Q^1$, $\{x^2\} = \{x^1\} \equiv \{x\}$, and the implication in $\{y^2\}$ of any $x \in \{x\}$ is the same as the implication in $\{y^1\}$ of that x.

As an example, any computer is a copy of itself. More generally, if V is a bijection over $\hat{U}$, then {X(V(.)), Q(V), Y(V(.))} is a copy of {X(.), Q, Y(.)}, where $Q(V) \equiv \{q(V(.)) : q \in Q\}$. Note that $Q^2 = Q^1$ means that $\{y^2\} = \{y^1\}$. An obvious generalization of Def. 2(v) is to only require that there be a re-ordering of the individual $q^2 \in Q^2$ and/or a bijective transformation of some of the $A(q^2 \in Q^2)$ such that $Q^2 = Q^1$.

Note that $X^1(.)$ may differ from $X^2(.)$ and that $Y^1(.)$ may differ from $Y^2(.)$ in the definition of a copy of a computer; the two computers are allowed to have different input values for the same $\hat{u}$, and they are allowed to have different output values for the same $\hat{u}$. (If this weren't the case, the two computers would be identical.) Similarly, they can have different $\hat{u}$ for the same output values (and/or input value). Accordingly, a particular partition can be weakly predictable to a computer C but not to a copy of C. (For example, this can occur when that partition is related to the output section of C's copy.)

It is possible to generalize Def. 2(v) so that C^1 and C^2 do not concern the same $\hat{U}$. The only place in our definition that the sharing of $\hat{U}$ arises is in the requirement that $Q^2 = Q^1$. To circumvent that requirement, given any countable set of partitions $\{\pi^i\}$, define $\Pi(\{\pi^i\})$ as the union over all $\hat{u}$ of the strings $(\pi^1(\hat{u}), \pi^2(\hat{u}), ...)$. (Since $\{\pi^i\}$is countable, so is $\Pi(\{\pi^i\})$.) This union is how the partitions collectively divide up $\hat{U}$. Then if we replace the requirement that $Q^2 = Q^1$ with the requirement that $\Pi(Q^2) = \Pi(Q^1)$, we arrive at our desired generalization.

If there is additional structure in the two $\hat{U}$ at hand, one can refine this generalization of the definition of a copy. For example, if both $\hat{U}$ are topological spaces that are homeomorphically related, one can require that the transformation implicit in establishing that $\Pi(Q^1) = \Pi(Q^2)$ respects that homeomorphism.

Definition 3: Consider a physical computer $C \equiv (Q, X(.), Y(.))$ and a $\hat{U}$-partition π. A (not necessarily surjective) partition mapping $\hat{U}$ into **B**, f, is an **intelligibility function** (for π) if

$$\forall\ \hat{u}, \hat{u}' \in \hat{U}, \pi(\hat{u}) = \pi(\hat{u}') \Rightarrow f(\hat{u}) = f(\hat{u}').$$

A set F of such intelligibility functions is an **intelligibility set** for π.

We view any intelligibility function as a question by defining A(f) to be the image of $\hat{U}$ under f. If F is an intelligibility set for π and $F \subseteq Q$, we say that π is **intelligible** to C with respect to F. If the intelligibility set is not specified, it is implicitly understood to be the set of all intelligibility functions for π.

We say that two physical computers C^1 and C^2 are **mutually intelligible** (with respect to the pair (F^1, F^2)) iff both Y^2 is intelligible to C^1 with respect to F^2 and Y^1 is intelligible to C^2 with respect to F^1.

Plugging in, π is intelligible to C iff $\forall$ intelligibility functions f, $\exists$ $q \in \{y_q\}$ such that q = f, i.e., such that A(q) = the image of $\hat{U}$ under f, and such that $\forall\ \hat{u} \in \hat{U}$, $q(\hat{u}) = f(\hat{u})$. Formally, by the surjectivity of Y(.), demanding intelligibility implies that $\exists\ \hat{u}' \in \hat{U}$ such that $\forall\ \hat{u} \in \hat{U}$, $[Y_q(\hat{u}')](\hat{u}) = f(\hat{u})$. Note that since π contains at least two elements, if π is intelligible to C, $\exists$ $y_q \in \{y_q\}$ such that $A(y_q) = \mathbf{B}$, an y_q such that $A(y_q) = \{0\}$, and one such that $A(y_q) = \{1\}$. Usually we are interested in the case where π is an output partition of a physical computer, as in mutual intelligibility.

In conventional computation as in Ex. 1, X(.) specifies the question $q \in Q$ we want to pose to the computer. In such scenarios, mutual intelligibility restricts how much computation can be "hidden" in $Y^2(.)$ and $X^1(.)$ ($Y^1(.)$ and $X^2(.)$, respectively) by coupling them, so that subsets of the range of $Y^2(.)$ are, directly, elements in the range of $X^1(.)$, without any intervening computational processing.

We are now in a position to formally define what it means for a computer to make a prediction. First consider the following three conditions relating a computer C, a partition π, and an intelligibility set for π, F:

i) π is intelligible to C with respect to F, i.e., $F \subseteq \{y_q\}$;

ii) $\forall\ f \in F$, $\exists\ x \in \{x\}$ that **weakly induces** f, i.e., an x such that:

$$X(\hat{u}) = x \;\Rightarrow\; Y_p(\hat{u}) = (A(f), f(\hat{u}));$$

iii) $\forall\ f \in F$, if the set of x values weakly inducing f is non-empty, then there is at least one of those x for which it is further true that $X(\hat{u}) = x \;\Rightarrow\; Y_q(\hat{u}) = f$.

Intuitively, condition (ii) means that for all questions q in F, there is an input state such that if C is initialized to that input state, C's answer to that question q (as evaluated at τ) must be correct. If (ii) and (iii) both hold, then we can combine those conditions into the single statement that $\forall$ f $\in$ F $\exists$ x $\in$ {x} such that $X(\hat{u}) = x \Rightarrow Y(\hat{u}) = (f, f(\hat{u}))$, and (i) is superfluous. Intuitively, in such a situation, for any question in the intelligibility set, there is always an input that induces the computer to ask and (correctly) answer that question.

Some of the unpredictability results do not require that all three conditions hold. In particular, our central result, Thm. 2, relies on neither (i) nor (iii); in its strongest formulation it only invokes condition (ii) (as the proof of it presented below makes clear). In contrast, existence proofs are strongest when we impose as many conditions as possible. This raises the issue of which of those conditions would most usefully be incorporated into our definition of predictability. As a compromise, here the term "weak predictability" is interpreted to mean only that conditions (i) and (ii) necessarily hold:

Definition 4: Consider a physical computer C, partition π, and intelligibility set for π, F. We say that π is **weakly predictable** to C with respect to F iff $F \subseteq \{y_q\}$, and $\forall$ f $\in$ F, $\exists$ x $\in$ {x} that weakly induces f.

As a formal matter, note that in the definition of predictable, even though f(.) is surjective onto A(f) (cf. Def. 3), it may be that for some x, the set of values $f(\hat{u})$ takes on when $\hat{u}$ is restricted so that $X(\hat{u}) = x$ do not cover all of A(f). The reader should also bear in mind that by surjectivity, $\forall$ x $\in$ {x}, $\exists\ \hat{u} \in \hat{U}$ such that $X(\hat{u}) = x$.

We next define the property that two computers' input functions are independent:

Definition 5: Consider a set of n physical computers $\{C^i \equiv (Q^i, X^i(.), Y^i(.)) : i = 1, ..., n\}$. We say $\{C^i\}$ is **(input) distinguishable** iff $\forall$ n-tuples $(x^1 \in \{x^1\}, ..., x^n \in \{x^n\})$, $\exists\ \hat{u} \in \hat{U}$ such that $\forall$ i,

$X^i(\hat{u}) = x^i$ simultaneously.

We say that $\{C^i\}$ is **pairwise (input) distinguishable** if any pair of computers from $\{C^i\}$ is distinguishable, and will sometimes say that any two such computers C^1 and C^2 "are distinguishable from each other". We will also say that $\{C^i\}$ is a **maximal** (pairwise) distinguishable set if there are no physical computers $C \notin \{C^i\}$ such that $C \cup \{C^i\}$ is a (pairwise) distinguishable set.

Our first result does not even concern the accuracy of prediction. It simply states that for any pair of physical computers there are *always* binary-valued questions about the state of the universe that cannot even be posed to at least one of those physical computers. In particular, this is true if the second computer is a copy of the first one, or even if it is the same as the first one. (The result does not rely on input-distinguishability of the two computers — a property that obviously does not describe the relationship between a computer and itself.) This impossibility holds no matter what the cardinality of the set of questions that can be posed to the computers (i.e., no matter what the cardinality of $\{x\}$ and/or Q). It is also true no matter how powerful the computers (and in particular holds even if the computers are more powerful than a Turing Machine), whether the computers are analog or digital, whether the universe is classical or quantum-mechanical, whether or not the computers are quantum computers, and even whether the computers are subject to physical constraints like the speed of light. In addition the result does not rely on chaotic dynamics in any manner. All that is required is that the universe contain two (perhaps identical, perhaps wildly different) physical computers.

Theorem 1: Consider any pair of physical computers $\{C^i : i = 1, 2\}$. Either $\exists$ finite intelligibility set F^2 for C^2 such that C^2 is not intelligible to C^1 with respect to F^2, and/or $\exists$ finite intelligibility set F^1 for C^1 such that C^1 is not intelligible to C^2 with respect to F^1.

Proof: Hypothesize that the theorem is false. Then C^1 and C^2 are mutually intelligible $\forall$ finite F^1 and F^2. Now the set of all finite F^2 includes any and all intelligibility functions for C^2, i.e., any and all functions taking $\hat{u}$ to a bit whose value is set by the value $Y^2(\hat{u})$. The set of those functions can be bijectively mapped to the power set $2^{\{y^2\}}$. So $F^2 \subseteq Q^1 \Rightarrow o(Q^1) \geq o(2^{\{y^2\}})$. However $o(\{y^2\}) \geq o(Q^2)$, since $\{y^2\}$ contains all possible specifications of a $q^2 \in Q^2$. Therefore $o(Q^1) \geq o(2^{Q^2})$. But it is always true that $o(2^A) > o(A)$ for any set A, which means in particular that $o(2^{Q^2}) > o(Q^2)$. Accordingly, $o(Q^1) > o(Q^2)$. Similarly though, $o(Q^2) > o(Q^1)$. Therefore $o(Q^1) > o(Q^1)$, which is impossible. **QED**.

Note that Thm. 1 doesn't require that C^1 and C^2 be different computers.

Ultimately, Thm. 1 holds due to our requiring that our physical computer be capable of answering more than one question about the future state of the universe. To satisfy this requirement q cannot be pre-fixed. (In conventional computation, it is specified in the computer's input.) But precisely because q is not fixed, for the computer's output of α to be meaningful it must be accompanied by specification of q; the computer's output must be a well-defined region in $\hat{U}$. It is this need to specify q as well as α in the output, ultimately, which means that one cannot have two physical computers both capable of being asked arbitrary questions concerning the output of the other.

Thm. 1 reflects the fact that while we do not want to have C's output partition "rigged ahead of time" in favor of some single question, we also cannot require too much flexibility of our computer. It is necessary to balance these two considerations before analyzing prediction of the future. We do this with the formal property of question-independence.

Recall that for any f that is an intelligibility function of (the output partition of) some computer C, $\forall\, \hat{u}, \hat{u}' \in \hat{U}$, $Y(\hat{u}) = Y(\hat{u}')$ implies that $f(\hat{u}) = f(\hat{u}')$. So for such an f, the joint condition $[Y_q(\hat{u}) = Y_q(\hat{u}')] \wedge [Y_\alpha(\hat{u}) = Y_\alpha(\hat{u}')]$ implies that $f(\hat{u}) = f(\hat{u}')$. We consider f's that obey weaker conditions:

Definition 6: An intelligibility function f for an output partition Y(.) is **question-independent** iff $\forall\, \hat{u}, \hat{u}' \in \hat{U}$:

$$Y_p(\hat{u}) = Y_p(\hat{u}') \Rightarrow f(\hat{u}) = f(\hat{u}').$$

An intelligibility set as a whole is question-independent if all its elements are.

We write $C^1 > C^2$ (or equivalently $C^2 < C^1$) and say simply that C^2 is **(weakly) predictable** to C^1 (or equivalently that C^1 **can predict** C^2) if $Y^2{}_p$ is weakly predictable to C^1 for all question-independent finite intelligibility sets for C^2. Similarly, from now on we will say that C^2 is **intelligible** to C^1 without specification of an intelligibility set if $Y^2{}_p$ is intelligible to C^1 with respect to all question-independent finite intelligibility sets for C^2.

Intuitively, f is question-independent if its value does not vary with q among any set of q all of which share the same A(q). As an example, say our physical computer is a conventional digital workstation. Have a certain section of the workstation's RAM be designated the "output section" of that workstation. That output section is further divided into a "question subsection" designating (i.e., "containing") a q, and an "answer subsection" designating an α. Say that for all q that can be designated by the question subsection A(q) is a single bit, i.e., we are only interested in binary-valued questions. Then for a question-independent f, the value of f can only depend on whether the answer subsection contains a 0 or a 1. It cannot vary with the contents of the question subsection. In terms of the first of the motivations we introduced for requiring intelligibility, requiring question-independent intelligibility means we only require each computer's *answer* to be readily intelligible to the other one. We are willing to forego having the question that each computer thinks it's answering also be readily intelligible to the other one.

As a formal example of question-independent intelligibility, say our computer has questions q for which A(q) = **B**, questions q for which A(q) = {0}, and q for which A(q) = {1}, but no others. Then there are four distinct subsets of $\hat{U}$, which mutually cover $\hat{U}$, defined by the four equations $Y_p(\hat{u}) = (\mathbf{B}, 1)$, $Y_p(\hat{u}) = (\mathbf{B}, 0)$, $Y_p(\hat{u}) = (\{1\}, 1)$, and $Y_p(\hat{u}) = (\{0\}, 0)$. (The full partition Y(.) is a refinement of this 4-way partition, whereas this 4-way partition need not have no relation with the partitions making up each q in Q.) So a question-independent intelligibility function of our computer is any **B**-valued function of which of these four subsets a particular $\hat{u}$ falls into.

Thm. 1 does not hold if we restrict attention to question-independent intelligibility sets. As an example, both of our computers could have their output answer subsections be a single bit, and both could have their Q contain all four Boolean questions about the state of the other computer's output answer bit. (Those are the following functions from $\hat{u} \in \hat{U} \rightarrow \mathbf{B}$: Is u such that the other computer's output bit is 1? 0? 1 and/or 0? Neither 1 nor 0?) So the Q of both computers contains all possible question-independent intelligibility sets for the other computer.

So Def. 6 allows us to circumvent Thm. 1. As an alternative solution, we could define a **question-free computer** as a pair of an input partition and an output partition where each output value y only consists of $A(y_q)$ and α (rather than y_q, $A(y_q)$, and α). Working with such computers would have the benefit of simplifying the analysis. Intelligibility in the sense originally defined, applied to a question-free computer, is exactly equivalent to applying question-independent intelligibility to a full (question-dependent) computer. Moreover many of the results of this paper still hold for question-free computers.

The problem with this alternative approach is that the two partitions X(.) and $Y_p(.)$, by themselves, don't really specify a "computer" in any sense. They don't specify a means of associating answers with questions. To address this without introducing Y_q, one might add a mapping from questions to inputs to the definition of a computer. However once one does this it is not clear that this new definition of a computer is any "simpler" than our original one. This approach is not pursued any further in this paper.

In general, we cannot have the x value of our computer C always uniquely fix the associated y_α (i.e., cannot have the case that $\forall$ x, $\exists$ y_α such that $X(\hat{u}) = x \Rightarrow Y_\alpha(\hat{u}) = y_\alpha$). If it did, then C could not predict most non-trivial computers that are distinguishable from C. For example, say that for a computer C^2, $\forall\ y^2{}_q \in \{y^2{}_q\}$, $A(y^2{}_q) = \{x^2\}$, and that $Y^2{}_\alpha(\hat{u}) = X^2(\hat{u})\ \forall\ \hat{u}$. So C^2's output simply equals its input. Then since *whatever* the choice of x all x^2 values are allowed (by distinguishability), it follows that whatever the choice of x, all $y^2{}_\alpha$ values are allowed. So appropriate choice of x cannot make the value y_α track (an intelligibility function of) $y^2{}_\alpha$ if that choice of x forces a unique value y_α.

This is quite reasonable. If C^1 is to predict C^2 correctly, the information of what C^2 is calculating must somehow be conveyed into C^1. Due to input-distinguishability, this can only happen by C^1's implicitly "observing" what question C^2 is answering (rather than by having x^1 reflect x^2). Accordingly, for a fixed x^1, C^1 must be able to generate different predictions, depending on the results of that "observing". Hence, x^1 cannot fix the value $y^1{}_\alpha$. (On the other hand, it is not so unreasonable to demand that the value of x^1 specify the value $y^1{}_q$, i.e., demand that it uniquely fixes what question C^1 is answering. See Coroll. 1 below.)

The following example establishes that there are pairs of input-distinguishable physical computers $\{C^1, C^2\}$ in which C^2 is predictable to C^1, and in which the *question* component of y^1 is uniquely fixed by x^1:

Example 2: Q^2 consists of a single question, one which is a binary partition of $\hat{U}$ so that $A(Y^2{}_q(\hat{u})) = \mathbf{B}$ always. Since $Y^2(.)$ is surjective, the image of $\hat{U}$ under $Y^2{}_\alpha(.)$ is all of $\mathbf{B}$. Q^1 has four elements given by the four logical functions of the bit $Y^2{}_\alpha(\hat{u})$. (Note these are the four intelligibility functions for C^2.) Have $X^1(.) = Y^1{}_q(.)$, so that $\{x^1\}$ contains four elements corresponding to those four possible questions concerning $y^2{}_\alpha$. Next, have $Y^1{}_\alpha(\hat{u}) = [Y^1{}_q(\hat{u})](\hat{u})\ \forall\ \hat{u} \in \hat{U}$. Then for any of the four intelligibility functions for C^2, q, $\exists\ x^1 \in \{x^1\}$ such that $X^1(\hat{u}) = x^1 \Rightarrow [A(Y^1{}_q(\hat{u})) = A(q)] \wedge [Y^1{}_\alpha(\hat{u}) = q(\hat{u})]$; simply choose $x^1 = q$, so that $X^1(\hat{u}) = x^1 \Rightarrow Y^1{}_q(\hat{u}) = q$. Finally, to ensure distinguishability, if there are multiple x^2 values, have each one occur for at least one $\hat{u}$ in each of the subregions of $\hat{U}$ given by the partition $X^1(.)$.

Due to question-independence, we do not need to specify $Y^2{}_q(.)$. If we like, we could set it so that $y^2{}_q$ is uniquely fixed by the value of x^2, just as is the case for C^1.

To ensure surjectivity of $Y^1(.)$, we could have $X^1(.)$ subdivide each of the two sets (one set for each value $y^2{}_\alpha$) $\{\hat{u} \in \hat{U} : Y^2{}_\alpha(\hat{u}) = y^2{}_\alpha\}$ into four non-empty subregions, one for each x^1 value. So $(X^1(\hat{u}), Y^2{}_\alpha(\hat{u}))$ are two-dimensional coordinates of a set of disjoint regions that form a rectangular array covering $\hat{U}$. This means that $\hat{u} \rightarrow (X^1(\hat{u}), Y^2{}_\alpha(\hat{u}))$ is surjective onto $\{x^1\} \times$

$\{y^2_\alpha\}$, so that for any y^1_α and intelligibility function of C^2, q, there is always a value of x^1 that both induces the correct prediction for that function q and is consistent with that y^2_α.

The following variant of Ex. 2 establishes that we could have yet another computer C^3 that predicts C^2 but that is also distinguishable from C^1:

Example 2': Have $Q^3 = Q^1$, $\{x^3\} = \{x^1\}$, $Y^3_q(.) = X^3(.)$, $Y^3_\alpha(\hat{u}) = [Y^3_q(\hat{u})](\hat{u})\ \forall\ \hat{u} \in \hat{U}$, and have $X^3(.)$ subdivide $X^1(.)$ so that all four values of x^3 can occur with each value of x^1. In general, as we vary over all $\hat{u} \in \hat{U}$ and therefore over all (x^1, x^3) pairs, the pair of intelligibility function that C^1 is predicting will separately vary from those that C^3 is predicting, in such a way that all 2^4 pairs of intelligibility functions for C^2 are answered correctly for some $\hat{u} \in \hat{U}$.

In addition, we can have a computer C^4, distinguishable from both C^1 and C^2, where $C^4 > C^1$, so that $C^4 > C^1 > C^2$. We can do this either with $C^4 > C^2$ or not, as the following variant of Ex. 2 demonstrates:

Example 2'': Have $Y^4_q(.) = X^4(.)$, $Y^4_\alpha(\hat{u}) = [Y^4_q(\hat{u})](\hat{u})\ \forall\ \hat{u} \in \hat{U}$, and $\{x^4\} = \{y^4_q\}$ equals the set of all 2^4 question-independent intelligibility functions for C^1. (There are four possible y^1_p: $\{(\{0\}, 0), (\{1\}, 1), (\mathbf{B}, 0), (\mathbf{B}, 1)\}$.) Ensure surjectivity of $Y^4(.)$ by having each region of constant $Y^4_q(\hat{u})$ overlap each region of constant $Y^1_p(\hat{u})$.This establishes that $C^4 > C^1$. Distinguishability would then hold if $X^4(.)$ subdivides $X^1(.)$ so that all 16 values of x^4 can occur with each value of x^1.

In this setup, C^2 may or may not be predictable to C^4. To see how it may not be, consider the case where $\{x^2\}$ is a single element (so distinguishability with C^2 is never an issue). Have $X^4(.)$ be a refinement of $Y^2_\alpha(.)$, in that each x^4 value can only occur with one or the other of the two y^2_α values. So each x^4 value delineates a "horizontal strip" of constant $Y^2_\alpha(\hat{u})$, running across all four values of $X^1(\hat{u})$. (Since $X^1(\hat{u}) = Y^1_q(\hat{u})$, and $Y^1_\alpha(\hat{u}) = (Y^1_q(\hat{u}))(\hat{u})$, $Y^1_\alpha(\hat{u}) =$

$(X^1(\hat{u}))(\hat{u})$, so specifying the value of $X^1(\hat{u})$ specifies $Y^1{}_p(\hat{u})$, and each strip crosses all four $y^1{}_p$ values, as was stipulated above.)

Now choose the strip with $A(Y^4{}_q(\hat{u})) = A(X^4(\hat{u})) = \{0\}$ to have coordinate $Y^2{}_\alpha(\hat{u}) = 1$, and the strip with $A(Y^4{}_q(\hat{u})) = \{1\}$ to have coordinate $Y^2{}_\alpha(\hat{u}) = 0$. In the remaining fourteen strips, $Y^4{}_\alpha(\hat{u})$ is not constant, and therefore is not a single-valued intelligibility function of the associated (constant) value of $Y^2{}_p(\hat{u})$. In both of those two strips though, $Y^4{}_\alpha(\hat{u})$ is the opposite of $Y^2{}_\alpha(\hat{u})$. So no x^4 value induces the identity question-independent intelligibility function of C^2: $\hat{u} \to OUT^2{}_\alpha(\hat{u})$, i.e., no x^4 induces $Y^4{}_p(\hat{u}) = (\mathbf{B}, Y^2{}_\alpha(\hat{u}))$. Accordingly, C^4 does not predict C^2.

In other instances though, both C^2 and C^1 are predictable to C^4. To have this we need only subdivide $\{x^4\}$ and $\{y^4\}$ into two portions, $(\{x^4\}_A, \{y^4\}_A)$, and $(\{x^4\}_B, \{y^4\}_B)$, which divide $\hat{U}$ in two. The first of these portions is used for predictions concerning C^2, as in Ex. 2; each region of constant $X^4(\hat{u})$ is a subset of a region of constant $X^1(\hat{u})$ overlapping both $Y^2{}_\alpha(\hat{u})$. The second is used for predictions concerning C^1, as just above. It consists of horizontal strips extending over that part of $\hat{U}$ not taken up by the regions with $X^4(\hat{u}) \in \{x^4\}_A$. So $\{x^4\}_A = \{y^4{}_q\}_A$ contains four elements, and $\{x^4\}_B = \{y^4{}_q\}_B$ contains sixteen, which means that $\{x\} = \{y\}$ contains twenty elements, all told. Distinguishability is ensured by having x^4 take on all its possible values within any subset of $\hat{U}$ over which both $X^1(.)$ and $X^2(.)$ are constant.

We now present the proof of Thm. 2:

Proof of Thm. 2: Given $Y^1(.)$ and $Y^2(.)$, define the function $f^2(\hat{u})$ by:

$f^2(\hat{u}) = 1$ if $A(Y^1{}_q(\hat{u})) = \{0\}$;

$f^2(\hat{u}) = 0$ if $A(Y^1{}_q(\hat{u})) = \{1\}$;

$f^2(\hat{u}) = NOT[Y^1{}_\alpha(\hat{u})]$ if $A(Y^1{}_q(\hat{u})) = \mathbf{B}$; and

$f^2(\hat{u}) = 0$ otherwise.

Intuitively, this function is the negation of Y^1's answer when Y^1's question is contained in $\mathbf{B}$. Now $A(f^2) \in \{\{0\}, \{1\}, \mathbf{B}\}$, with its precise value depending on y^1. Since by construction f^2 does not vary with $Y^1_q(\hat{u})$, only with $A(Y^1_q(\hat{u}))$, this means that f^2 is a question-independent intelligibility function of Y^1. Define f^1 similarly, just with no negation operation; $f^1(\hat{u}) = Y^2_\alpha(\hat{u})$, whenever $A(y^2_q) \subseteq \mathbf{B}$, and equals 0 otherwise.

By hypothesis, $\exists\ x^2$ such that $X^2(\hat{u}) = x^2 \Rightarrow Y^2_p(\hat{u}) = (A(f^2), f^2(\hat{u}))$. (Note that for that x^2, $A(Y^2_q(\hat{u})) \in \{\{0\}, \{1\}, \mathbf{B}\}$.) Similarly for x^1 and f^1. So by input distinguishability, $\exists$ single $\hat{u}$ such that at the same time, $Y^2_\alpha(\hat{u}) = f^2(\hat{u})$ and $Y^1_\alpha(\hat{u}) = f^1(\hat{u})$. Plugging in and using the fact that both $A(Y^2_q(\hat{u})) \in \{\{0\}, \{1\}, \mathbf{B}\}$ and $A(Y^1_q(\hat{u})) \in \{\{0\}, \{1\}, \mathbf{B}\}$, we see that $Y^1_\alpha(\hat{u}) = f^1(\hat{u}) = Y^2_\alpha(\hat{u}) = f^2(\hat{u}) = \text{NOT}[Y^1_\alpha(\hat{u})]$. This contradiction establishes our result. **QED.**

Restating it, Thm. 2 says that either $\exists$ finite question-independent intelligibility set for C^1, F^1, such that C^1 is not predictable to C^2 with respect to F^1, and/or $\exists$ finite question-independent intelligibility set for C^2, F^2, such that C^2 is not predictable to C^1 with respect to F^2. We can weaken the definition of "intelligibility" and still establish the impossibility of having both $C^1 > C^2$ and $C^2 > C^1$. For example, that impossibility will still obtain even if neither C^1 nor C^2 contains $\mathbf{B}$-valued questions, if they instead contain all possible functions mapping each others' values of y_α onto {0, 1, 2} (or more precisely contain all such functions of y_p — cf. the definition of prediction partition). For pedagogical simplicity, such weakened definitions are not investigated here.

Note that Thm. 2 still holds if we consider larger intelligibility sets that are supersets of F, the set of all intelligibility functions of Y_p. In particular, consider modifying the definition of weak predictability to involve F', the set of all intelligibility functions of the partition $\hat{u} \rightarrow (X(\hat{u}), Y_p(\hat{u}))$. Intuitively, this is the set of all (question-independent) intelligibility functions of the entire computer (X, Y), not just of its output partition. (So "prediction" now means, in essence, predicting all aspects of C.) Then since $F \subseteq F'$, Thm. 2 still applies with this alternative definition of weak predictability.

As mentioned previously, Thm. 2 does not rely on intelligibility. This reflects our restriction to question-independent intelligibility functions. Such functions cannot "see" what the contents of some (computer-to-be-predicted's) y_q are. Similarly, condition (ii) does not care about the contents of any (predicting computer's) y_q. So the contents of y_q in either a predicting or being-predicted computer are, for the most part, irrelevant. Accordingly, restrictions on those contents have few effects concerning computers predicting each other using question-independent intelligibility sets.

Nonetheless, Thm. 2 can be used to derive an uncomputability result that does rely on mutually intelligibility. To see this, define a computer C to be (**Y_q**) **stable** if $\forall$ q $\in$ $\{y_q\}$, there is always an associated input that forces the output question to equal q, i.e., if $\exists$ x such that $X(\hat{u}) = x \Rightarrow Y_q(\hat{u}) = q$. (Note that given any y, stability can always be assured by choosing a sufficiently fine-grained X(.).) In addition, define a computer to be **infallible** if its associated answers are always correct responses to its associated questions, i.e., if $Y_\alpha(\hat{u}) = [Y_q(\hat{u})](\hat{u})\ \forall\ \hat{u}$. (As an example, given any partition π, the computer which has a single question given by $q(\hat{u}) = \pi(\hat{u})$ and which has $Y_\alpha(\hat{u}) = \pi(\hat{u})$ is infallible.) Then we have the following:

Corollary 1: Let C^1 and C^2 be two distinguishable mutually intelligible computers, both of which are stable. It is not possible that both C^1 and C^2 are infallible.

Proof: Let F^2 be the set of all questions-independent intelligibility functions for C^2. Then $F^2 \subseteq \{y^1{}_q\}$, by mutual intelligibility. By stability of Y^1, this means that $\forall$ f $\in$ F^2, $\exists$ x $\in$ $\{x_1\}$ such that $X_1(\hat{u}) = x \Rightarrow Y^1{}_q(\hat{u}) = f$. If C^1 were infallible, this would then mean that $Y^1{}_p(\hat{u}) = (A(f), f(\hat{u}))$. So x weakly induces f, and more generally, $C^1 > C^2$. Similarly, $C^2 > C^1$. If we now apply Thm. 2 we get the result claimed. **QED**.

Similarly, one can produce corollaries of the results presented below by, in essence, replacing predictability with infallibility. For reasons of space, those corollaries are not presented here. Note that for any stable, infallible computer C, if C' is intelligible to C, then all three conditions (i-iii) considered for defining weak predictability hold.

As an aside, there are several ways one can generalize the foregoing to the case of stochastic scenarios. One starts by defining a **probabilistic partition** R as a space of partition labels A(R) and an associated distribution $P_R(r \in A(R) \mid \hat{u})$. (The situation considered heretofore is the special case where all partitions are delta functions.) In particular, an output probabilistic partition Y is one where $\{A(y)\}$ is the set of all pairs $\{q \in \{y_q\}, \alpha \in A(q)\}$ for some set of probabilistic partitions $\{y_q\}$. An example is a workstation whose output answer α is the specification of one of a set of candidate Gaussian distributions concerning the external world, i.e., a Gaussian $P(\hat{u} \mid \alpha)$. Given also a prior distribution $P(\alpha)$, we can express that workstation's output as a probabilistic question $P(\alpha \mid \hat{u})$ together with a particular associated answer. Another example is where $\hat{u}$ is a wavefunction, and a probabilistic partition gives the results of a Hermitian operator applied to that wavefunction.

For simplicity assume that the full joint distribution over $\hat{U}$ and all partition labels is specified, and that $P(\hat{u})$ is nowhere-zero over its domain of definition. Now any actual physical computer's state is specified in $\hat{u}$ for a classical universe, and the same is true in the quantum case assuming $\hat{u}$ is an eigenstate of the operator of a human observing the computer's output. Accordingly, the input and output probabilistic partitions of a probabilistic computer (i.e., $P(x \in \{x\} \mid \hat{u})$ and $P(y \in \{y\} \mid \hat{u})$, respectively) are delta functions, although the partition Y_q is not one in general. Two probabilistic computers C^1 and C^2 are (input) **probabilistic distinguishable** if $\forall\ x^1 \in \{x^1\}$ and $x^2 \in \{x^2\}$, $\exists\ \hat{u}$ such that $P(\hat{u}) \neq 0$, $P(x^1 \mid \hat{u}) \neq 0$, and $P(x^2 \mid \hat{u}) \neq 0$.

As before, an intelligibility function is a "translation" mapping a partition's possible outputs into **B**. Formally, a **probabilistic intelligibility function** Φ of a (probabilistic) partition R with labels r is a probabilistic partition having $A(\Phi) \subseteq \mathbf{B}$ where $\exists$ a single-valued function h: $R \rightarrow \mathbf{B}$ such that $P(\phi \in A(\Phi) \mid \hat{u}) = \int d\hat{u}\ \delta(\phi, h(r))\ P(r \mid \hat{u})$. (A question-independent probabilistic intelligibility function of an output partition Y simply has h(y) depend only on y_p.) We define the **degree of weak predictability** of a probabilistic partition R to a probabilistic computer C for an intelligibility set F as

$$\varepsilon_{R:C} = min_{\Phi \in F} max_{IN} \int d\hat{u}\ P_{IN}(\hat{u}|IN) \sum_{\phi, b} \delta(\phi, b)\ P_{\Phi}(\phi|\hat{u})\ P_{OUT}(OUT_p = (A(f), b)|\hat{u})\ .$$

Intuitively, this is the minimax probability of C’s answer (b) agreeing with Φ’s answer (ϕ).

Note that $\varepsilon_{R:C} = 1$ implies that $\phi = b\ \forall\ \hat{u}$ such that $P(\hat{u} \mid x)$ is non-zero (for the maximizing x). Now since output partitions are delta functions, if R is the output partition of a computer C', then all $\Phi \subseteq F$ are delta functions. In other words, those intelligibility functions are single-valued functions from $\hat{U}$ to **B** (as always are the partitions X and Y). Accordingly, having ϕ necessarily equal b reduces to the conventional (non-probabilistic) definition of weak predictability, and Thm. 2 applies. This proves that it is impossible to have two distinguishable probabilistic computers C^1 and C^2 such that $\varepsilon_{C^1:C^2} = \varepsilon_{C^2:C^1} = 1$.

Returning to the case of non-probabilistic partitions, we now present a result that is often handy in working with systems meeting our definition of weak predictability (i.e., conditions (i) and (ii)). First note that for any partition π containing at least two elements, there exists an intelligibility function f for π with A(f) = **B**, an intelligibility function f with A(f) = {1}, and an intelligibility function f with A(f) = {0}. By exploiting the surjectivity of output partitions, we can extend this result to concern such partitions. This is formally established in the following lemma, which holds whether or not we assume partitions are binary:

Lemma 1: Consider a physical computer C^1. If ∃ any output partition Y^2 that is intelligible to C^1, then $\exists\ q^1 \in Q^1$ such that $A(q^1) = \mathbf{B}$, a $q^1 \in Q^1$ such that $A(q^1) = \{0\}$, and a $q^1 \in Q^1$ such that $A(q^1) = \{1\}$.

Proof: Since $\{y^2\}$ is non-empty, $\{y^2{}_q\}$ is non-empty. Pick some $q^* \in \{y^2{}_q\}$ having at least two elements. (By definition of physical computer, there is at least one such q^*.) Construct any binary-valued function f^{*2} of $\alpha \in A(q^*)$ such that there exists at least one α for which $f^{*2}(\alpha) = 0$ and at least one for which $f^{*2}(\alpha) = 1$. Define an associated function $f^{*2}(\hat{u}) = f^{*2}(Y^2_\alpha(\hat{u}))$ if $A(Y^2_q(\hat{u})) = A(q^*)$, 0 otherwise. By the surjectivity of $Y^2(.)$, $\forall\ \alpha \in A(q^*)$, $\exists\ \hat{u}$ such that both $Y^2_q(\hat{u}) = q^*$ and $Y^2_\alpha(\hat{u}) = \alpha$. Therefore $\exists\ \hat{u}$ such that $f^{*2}(\hat{u}) = 1$, and $\exists\ \hat{u}$ such that $f^{*2}(\hat{u}) = 0$. This establishes, by construction, that there is a question-independent intelligibility function of C^2 that takes on both the value 1 and the value 0, f^{*2}. So by our hypothesis that C^2 is intelligible to C^1 with respect to any question-independent intelligibility function of C^2, we know that $f^{*2} \in Q^1$. Moreover, viewed as a question, $A(f^{*2}) = \mathbf{B}$. So, we have established that Q^1 contains a binary valued function.

Next, note that the function $\hat{u} \in \hat{U} \rightarrow 1$ is always a question-independent intelligibility function of C^2, as is the function $\hat{u} \in \hat{U} \rightarrow 0$. Again using surjectivity, we see that A for these two functions are $\{1\}$ and $\{0\}$, respectively. **QED.**

We now present proofs of some other results presented in the main text.

Proof of Coroll. 2: Hypothesize that the corollary is wrong. Define the composite device $C^* \equiv (IN^*(.) \equiv \Pi_{i=1}^{n-1}\ X^i(.), Q^1, Y^1(.))$. Since $\{C^i\}$ is fully distinguishable, $X^*(.)$ is surjective. Therefore C^* is a physical computer.

Since by hypothesis C^n is intelligible to C^{n-1}, $\exists\ y^{n-1}_q$ such that $A(y^{n-1}_q) = \mathbf{B}$. Also, since $C^{n-2} > C^{n-1}$, $\exists\ x^{n-2} \in \{x^{n-2}\}$ such that $\forall\ \hat{u} \in \hat{U}$ for which $A(Y^{n-1}_q(\hat{u})) = \mathbf{B}$, $X^{n-2}(\hat{u}) = x^{n-2} \Rightarrow y^{n-2}_\alpha(\hat{u}) = Y^{n-1}_\alpha(\hat{u})$. Iterating and exploiting full distinguishability, $\exists\ (x^1, ..., x^{n-2})$ such that $\forall\ \hat{u} \in \hat{U}$ for which $A(Y^{n-1}_q(\hat{u})) = \mathbf{B}$, $(X^1(\hat{u}), .., X^{n-2}(\hat{u})) = (x^1, ..., x^{n-2}) \Rightarrow Y^*(\hat{u}) = Y^1(\hat{u}) = Y^{n-1}(\hat{u})$. The same holds when we restrict $\hat{u}$ so that the space $A(Y^{n-1}_q(\hat{u})) = \{1\}$, and when we restrict $\hat{u}$ so that $A(Y^{n-1}_q(\hat{u})) = \{0\}$.

Since by hypothesis C^n is intelligible to C^{n-1}, and since $X^*(.)$ is surjective, this result means that C^n is predictable to C^*. Conversely, since $C^n > C^1$ by hypothesis, the output partition of C^* is predictable to C^n, and therefore C^* is. Finally, since $\{C^i\}$ is fully distinguishable, C^* and C^n are distinguishable. Therefore Thm. 2 applies, and by using our hypothesis we arrive at a contradiction. **QED.**

Proof of Thm. 3: Assume our corollary is wrong, and some computer C is predictable to itself. Since by definition predictability implies intelligibility, we can apply Lemma 1 to establish that there is a $q \in \{y_q\}$, q', such that A(q') = **B**. Therefore one question-independent intelligibility function of C is the function f from $\hat{u} \in \hat{U} \to \mathbf{B}$ that equals 1 if $A(Y_q(\hat{u})) = \mathbf{B}$ and $Y_\alpha(\hat{u}) = 0$, and equals 0 otherwise. Therefore by hypothesis $\exists\, x \in \{x\}$ such that $X(\hat{u}) = x \Rightarrow A(Y_q(\hat{u})) = \mathbf{B}$ and $Y_\alpha(\hat{u}) = f(\hat{u})$. But if $\{A(Y_q(\hat{u}))\} = \mathbf{B}$, then $f(\hat{u}) = \mathrm{NOT}[Y_\alpha(\hat{u})]$, by definition of f(.). Since X is surjective, this means that there is at least one $\hat{u} \in \hat{U}$ such that $\{A(Y_q(\hat{u}))\} = \mathbf{B}$ and $Y_\alpha(\hat{u}) = \mathrm{NOT}[Y_\alpha(\hat{u})]$. This is impossible. **QED.**

For analyzing god computers the following definition is useful:

Definition 7: Consider a pairwise distinguishable set $\{C^i\}$ with god computer C^1. Define the partitions $Y^{i\times j}(\hat{u} \in \hat{U}) \equiv (Y_q^{i\times j}(\hat{u}), Y_\alpha^{i\times j}(\hat{u}))$, where each answer map $Y_\alpha^{i\times j}(\hat{u}) \equiv (Y^1{}_\alpha(\hat{u}), Y^2{}_\alpha(\hat{u}))$, and each question $[Y_q^{i\times j}(\hat{u})] \equiv$ the mapping given by $\hat{u}' \in \hat{U} \to ([Y^1{}_q(\hat{u})](\hat{u}'), [Y^2{}_q(\hat{u})](\hat{u}'))$. Then C^1 is **omniscient** if $Y^{2\times 3\times \ldots}$ is weakly predictable to C^1.

Intuitively, $Y^{i\times j}$ is just the double partition $(Y^i(.), Y^j(.)) = ((Y^i{}_q(.), Y^i{}_\alpha(.)), (Y^j{}_q(.), Y^j{}_\alpha(.))$, re-expressed to be in terms of a single question-valued partition and a single answer-valued partition. To motivate this re-expression, for any two questions $q^i \in Q^i$ and $q^j \in Q^j$, let $q^i \times q^j$ be the ordered product of the partitions q^i and q^j; it is the partition assigning to every point $\hat{u}' \in \hat{U}$ the label $(q^i(\hat{u}'), q^j(\hat{u}'))$. Then if $Y^i{}_q(\hat{u})$ is the question q^i and $Y^j{}_q(\hat{u})$ is the question q^j, $Y_q^{i\times j}(\hat{u})$ is the question $q^i \times q^j$. $Y_\alpha^{i\times j}$ is defined similarly, only with one fewer levels of "indirection", since answer components of output partitions are not themselves partitions (unlike question components). Note that even though any $Y^i(.)$ and $Y^j(.)$ are both surjective mappings, $Y^{i\times j}$ need not be surjective onto the set of quadruples $\{q^i \in Q^i, q^j \in Q^j, \alpha^i \in A(Q^i), \alpha^j \in A(Q^j)\}$.

Corollary 3: Consider three pair-wise distinguishable computers C^1, C^2, C^3, where $\nexists\, q^3 \in Q^3$ such that $A(q^3) \nsubseteq \mathbf{B}$. Assume that C^1 is an omniscient computer, and that C^1 is intelligible to C^3. Finally, assume further that not only that C^3's output can be any of its possible question-answer pairs, but also that for any of its questions, for any of the associated possible answers, there are situations where that answer is correct (so that C^2 should leave C^3's answer alone in those situations). (Formally, this means that $\forall$ pairs $(q^3 \in Q^3, \alpha^3 \in A(q^3))$, $\exists\, \hat{u} \in \hat{U}$ such that both $Y^3_q(\hat{u}) = q^3$ and $q^3(\hat{u}) = \alpha^3$, i.e., $[Y^3_q(\hat{u})](\hat{u}) = \alpha^3$.) Then it is not possible that $\forall\ \hat{u} \in \hat{U}$, $Y^2_\alpha(\hat{u}) = 1$ if $[Y^3_q(\hat{u})](\hat{u}) = Y^3_\alpha(\hat{u})$, 0 otherwise.

Proof: Hypothesize that the corollary is wrong. Construct a composite device $C^{2\text{-}3}$, starting by having $X^{2\text{-}3}(.) \equiv Y^3_q(.)$, $Q^{2\text{-}3} = Q^3$ and $Y^{2\text{-}3}_q(.) = Y^3_q(.)$. Next define the question θ by the rule $\theta(\hat{u}) \equiv \text{NOT}[Y^3_\alpha(\hat{u})]$ if $Y^2_\alpha(\hat{u}) = 0$, $\theta(\hat{u}) \equiv Y^3_\alpha(\hat{u})$ otherwise. (N.b. no assumption is made that $\theta \in Q^{2\text{-}3}$.) To complete the definition of the composite computer $C^{2\text{-}3}$, have $Y^{2\text{-}3}_\alpha(\hat{u}) = \theta(\hat{u})$.

Now by our hypothesis, $\forall\ \hat{u} \in \hat{U}$, $\theta(\hat{u}) = [Y^3_q(\hat{u})](\hat{u})$. By the last of the conditions specified in the corollary, this means that $\forall\ (q^{2\text{-}3} \in Q^{2\text{-}3}, \alpha^{2\text{-}3} \in A(q^{2\text{-}3}))$, $\exists\, \hat{u}$ such that $Y^{2\text{-}3}_q(\hat{u}) = q^{2\text{-}3}$ and $Y^{2\text{-}3}_\alpha(\hat{u}) = \alpha^{2\text{-}3}$. So $C^{2\text{-}3}$ allows all possible values of $\{y^{2\text{-}3}\}$, as a physical computer must. Due to surjectivity of Y^3_q, it also allows all possible values of the space $\{x^{2\text{-}3}\}$. To complete the proof that $C^{2\text{-}3}$ is a (surjective) physical computer, we must establish that $Y^{2\text{-}3}_\alpha(\hat{u}) \in A(Y^{2\text{-}3}_q(\hat{u}))\ \forall\ \hat{u} \in \hat{U}$. To do this note that if for example $A(Y^{2\text{-}3}_q(\hat{u})) = A(Y^3_q(\hat{u})) = \{1\}$, then since it is always the case that the $Y^{2\text{-}3}_\alpha(\hat{u}) = [Y^{2\text{-}3}_q(\hat{u})](\hat{u}) = [Y^3_q(\hat{u})](\hat{u})$, $Y^{2\text{-}3}_\alpha(\hat{u}) = 1$. Similarly $Y^{2\text{-}3}_\alpha(\hat{u}) \in A(Y^{2\text{-}3}_q(\hat{u}))$ when $A(Y^{2\text{-}3}_q(\hat{u})) = \{0\}$. Finally, if $A(Y^{2\text{-}3}_q(\hat{u})) = \mathbf{B}$, then the simple fact that $Y^{2\text{-}3}_\alpha(\hat{u}) \in \mathbf{B}$ always means that $Y^{2\text{-}3}_\alpha(\hat{u}) \in A(Y^{2\text{-}3}_q(\hat{u}))$.

Since C^1 is intelligible to C^3 and $Q^{2\text{-}3} = Q^3$, C^1 is intelligible to $C^{2\text{-}3}$. Moreover, given any question $q^{2\text{-}3} \in Q^{2\text{-}3}$, $\exists$ associated $x^{2\text{-}3} \in \{x^{2\text{-}3}\}$ such that $\forall\ \hat{u} \in \hat{U}$ for which $X^{2\text{-}3}(\hat{u}) = x^{2\text{-}3}$, $Y^{2\text{-}3}(\hat{u}) = q^{2\text{-}3}$. But as was just shown, $Y^{2\text{-}3}_\alpha(\hat{u}) = q^{2\text{-}3}(\hat{u})$ for that $\hat{u}$. Therefore C^1 is predictable to $C^{2\text{-}3}$.

Next, since C^1 is omniscient, $Y^{2\times3}$ is intelligible to C^1. Therefore any binary function of the regions defined by quadruples $(A(Y^2_q(\hat{u})), A(Y^3_q(\hat{u})), Y^2_\alpha(\hat{u}), Y^3_\alpha(\hat{u}))$ is an element of Q^1. Any single such region is wholly contained in one region defined by the pair $(A(Y^{2\text{-}3}_q(\hat{u})), Y^{2\text{-}3}_\alpha(\hat{u}))$ though. Therefore any binary function of the regions defined by such pairs is an element of Q^1. Therefore $C^{2\text{-}3}$ is intelligible to Q^1. Similarly, the value of any such binary function must be given by $Y^1_\alpha(\hat{u})$ whenever $X^1(\hat{u})$ equals some associated x^1. So $C^{2\text{-}3}$ is predictable to C^1.

Finally, since C^1 and C^3 are input-distinguishable, so are C^1 and $C^{2\text{-}3}$, and therefore Thm. 2 applies. This establishes that our hypothesis results in a contradiction. **QED.**

Similarly, we cannot arrange to have two computers be "anti-predictable" to one another. This is mentioned in the main text as Coroll. 4 of Thm. 2. The proof of this result is as follows:

Proof of Coroll. 4: By assumption C^1 and C^2 are mutually intelligible. So what we must establish is whether for both of them, for all intelligibility functions concerning the other one, there exists an appropriate value of x^i such that that intelligibility function is incorrectly predicted.

Hypothesize that the corollary is wrong. Then $\forall$ question-independent intelligibility functions for C^1, f^1, $\exists\ x^2 \in \{x^2\}$ such that $X^2(\hat{u}) = x^2$ implies that $[A(Y^2_q(\hat{u})) = \mathrm{NOT}[A(f^1)]] \wedge [Y^2_\alpha(\hat{u}) = \mathrm{NOT}[f^1(\hat{u})]]$. However by definition of question-independent intelligibility functions, given any such f^1, there must be another question-independent intelligibility function of C^1, f^3, defined by $f^3(.) \equiv \mathrm{NOT}(f^1(.))$. Therefore $\exists\ x^2 \in \{x^2\}$ such that $X^2(\hat{u}) = x^2$ implies that $[A(Y^2_q(\hat{u})) = A(f^3)] \wedge [Y^2_\alpha(\hat{u}) = f^3(\hat{u})]$.

This NOT(.) transformation bijectively maps the set of all question-independent intelligibility functions for C^2 onto itself. Since that set is finite, this means that the image of the set under the NOT(.) transformation is the set itself. Therefore our hypothesis means that all question-independent functions for C^1 can be predicted correctly by C^2 for appropriate choice of $x^2 \in \{x^2\}$. By similar reasoning, we see that C^1 can always predict C^2 correctly. Since C^1 and C^2 are distinguishable, we can now apply Thm. 2 and arrive at a contradiction. **QED.**

Recall that there are three conditions related to weak predictability, and for pedagogical simplicity we settled on two for our formal definition of the term (cf. discussion preceding Def. 4). The situation with strong predictability is closely analogous. Its formal definition involving two conditions is as follows:

Definition 8: Consider a pair of physical computers C^1 and C^2. We say that C^2 is **strongly predictable** to C^1 (or equivalently that C^1 **can strongly predict** C^2), and write $C^1 >> C^2$ (or equivalently $C^2 << C^1$) iff:

i) C^2 is intelligible to C^1;

ii) $\forall$ question-independent intelligibility functions for C^2, q^1, $\forall\ x^2 \in \{x^2\}$,

$\exists\ x^1 \in \{x^1\}$ that **strongly induces** the pair (q^1, x^2), i.e., such that:

$$X^1(\hat{u}) = x^1$$

$$\Rightarrow$$

$$[Y^1{}_p(\hat{u}) = (A(q^1), q^1(\hat{u}))] \ \wedge\ [X^2(\hat{u}) = x^2].$$

We now present the proofs of some of the fundamental theorems concerning strong predictability:

Proof of Thm. 5: To prove (i), let f be any question-independent intelligibility function of π. By Lemma 1, the everywhere 0-valued question-independent intelligibility function of π is contained in Q^1, and since $C^1 > C^2$, there must be an x^1 such that $X^1(\hat{u}) = x^1 \Rightarrow Y^1{}_\alpha(\hat{u}) = 0$. The same is true for the everywhere 1-valued function. Therefore to prove the claim we need only establish that for every question-independent intelligibility function of π, f, for which $A(f) = \mathbf{B}$, $f \in Q^1$, and there exists an x^1 such that $X^1(\hat{u}) = x^1 \Rightarrow Y^1{}_\alpha(\hat{u}) = f(\hat{u})$. Restrict attention to such f from now on.

Define a question-independent intelligibility function of C^2, I^2, such that $A(I^2) = \mathbf{B}$, and such that for all $\hat{u}$ for which $A(Y_q(\hat{u})) = \mathbf{B}$, $I^2(\hat{u}) = Y^2{}_\alpha(\hat{u})$. (Note that since $C^2 > \pi$, there both exist $\hat{u}$ for which $Y^2{}_p(\hat{u}) = (\mathbf{B}, 1)$ and $\hat{u}$ such that $Y^2{}_p(\hat{u}) = (\mathbf{B}, 0.)$ Now by hypothesis, for any of the f we are considering, $\exists\, x^2{}_f \in \{x^2\}$ such that $X^2(\hat{u}) = x^2{}_f \Rightarrow Y^2{}_p(\hat{u}) = (\mathbf{B}, f(\hat{u}))$. However the fact that $C^1 >> C^2 \Rightarrow \exists\, x^1 \in \{x^1\}$ such that $X^1(\hat{u}) = x^1 \Rightarrow X^2(\hat{u}) = x^2{}_f$ and such that $Y^1{}_p(\hat{u}) = (A(I^2), I^2(\hat{u})) = (\mathbf{B}, I^2(\hat{u}))$. Since $X^2(\hat{u}) = x^2{}_f$ for such a $\hat{u}$, $A(Y^2{}_\alpha(\hat{u})) = \mathbf{B}$, and therefore $I^2(\hat{u}) = Y^2{}_\alpha(\hat{u})$. So $Y^2{}_p(\hat{u})$ for such a $\hat{u}$ equals $(\mathbf{B}, Y^2{}_\alpha(\hat{u}))$. So for that x^1, $Y^1{}_p(\hat{u}) = (A(f), f(\hat{u}))$.

This establishes (i). The proof for (ii) goes similarly, with the redefinition that $x^1{}_f$ fixes the value of x^3 as well as ensuring that $Y^2{}_p(\hat{u}) = (A(f), f(\hat{u}))$. **QED.**

Proof of Thm. 6: Choose any x^2. For any question-independent intelligibility function of $y^2{}_p$, f, there must exist an $x^1{}_f \in \{x^1\}$ that strongly induces x^2 and f, since $C^1 >> C^2$. Label any such x^1 as $x^1{}_f$ (x^2 being implicitly fixed). So for any such f, $\{\hat{u} : X^1(\hat{u}) = x^1{}_f\} \subseteq \{\hat{u} : X^2(\hat{u}) = x^2\}$. However since $\{y^2{}_p\}$ is not empty, there are at least two question-independent intelligibility functions of $y^2{}_p$, f_1 and f_2, where $A(f_1) \neq A(f_2)$ (cf. Lemma 1). Moreover, the intersection $\{\hat{u} : X^1(\hat{u}) = x^1{}_{f_1}\} \cap \{\hat{u} : X^1(\hat{u}) = x^1{}_{f_2}\} = \varnothing$, since these two sets induce different $A(y^1{}_q)$ (namely $A(f_1)$ and $A(f_2)$, respectively). This means that $\{\hat{u} : X^1(\hat{u}) = x^1{}_{f_1}\} \subset \{\hat{u} : X^2(\hat{u}) = x^2\}$. On the other hand, for the same reasons, there must also exist an x^2 that strongly induces $x^1{}_{f_1}$. Therefore $\exists\, x^{2'}$ such that $\{\hat{u} : X^2(\hat{u}) = x^{2'}\} \subset \{\hat{u} : X^1(\hat{u}) = x^1{}_{f_1}\}$. So $\{\hat{u} : X^2(\hat{u}) = x^{2'}\} \subset \{\hat{u} : X^2(\hat{u}) = x^2\}$. This is not compatible with the fact that $X^2(.)$ is a partition. **QED.**

The following theorems involve physical computation analogues of TM theory.

Theorem 4: Given a set of physical computers $\{C^i\}$, $\nexists\, C^1 \in \{C^i\}$ such that $\forall\, C^2 \in \{C^i\}$,

i) C^2 is intelligible to C^1;

ii) $\forall\, q^2 \in Q^2, \exists\, x^1 \in \{x^1\}$ such that $X^1(\hat{u}) = x^1 \Rightarrow Y^1_\alpha(\hat{u}) = 1$ iff $q^2(\hat{u}) = Y^2_\alpha(\hat{u})$.

Proof: Choose C^2 such that $Y^2(.) = Y^1(.)$. (If need be, to do this simply choose $C^2 = C^1$.) Then in particular, $Y^1_\alpha(.) = Y^2_\alpha(.)$. Now since C^2 is intelligible to C^1 by hypothesis, by Lemma 1 $\exists\, q^1 \in Q^1$ such that $A(q^1) = \{0\}$, and therefore $\exists\, q^2 \in Q^2$ such that $A(q^2) = \{0\}$. For that q^2, $Y^1_\alpha(\hat{u}) = 1$ iff $0 = Y^1_\alpha(\hat{u})$, which is impossible. **QED.**

We now present definitions needed to analyze prediction complexity.

Definition 10: For any physical computer C with input space $\{x\}$:

i) Given any partition π, a **(weak) prediction input set** (of C, for π) is any set $s \subseteq \{x\}$ such that both every intelligibility function of π is weakly induced by an element of s, and for any proper subset of s at least one such function is not weakly induced. We write the space of all weak prediction input sets of C for π as $C^{-1}(\pi)$.

ii) Given any other physical computer C' with input space $\{x'\}$ for which the set of all question-independent intelligibility functions is $\{f'\}$, a (**strong**) **prediction input set** of C, for the triple C', $\Xi' \subseteq \{x'\}$, and $f' \subseteq \{f'\}$, is any set $s \subseteq \{x\}$ such that both every pair $(f' \in f', x' \in \Xi')$ is strongly induced by a member of s, and for any proper subset of s at least one such pair is not strongly induced. We write the space of all strong prediction input sets (of C, for C', Ξ', and f') as $C^{-1}(C', \Xi', f')$.

Definition 11: Given a physical computer C and a measure $d\mu$ over $\hat{U}$:

i) Define $V(\Xi \subseteq \{x\})$ as the measure of the set of all $\hat{u} \in \hat{U}$ such that $X(\hat{u}) \in \Xi$, and define the **length** of Ξ (with respect to X(.)) as $\mathbf{l}(\Xi) \equiv -\ln[V(\Xi)]$;

ii) Given a partition π that is predictable to a physical computer C, define the **prediction complexity** of π (with respect to C), $\mathbf{c}(\pi \mid C)$, as $\min_{\rho \in C^{-1}(\pi)} [\mathbf{l}(\rho)]$.

Proof of Thm. 7: Given any intelligibility function f for π, consider any $x^2{}_f \in \{x^2\}$ that weakly induces f, i.e., such that $X^2(\hat{u}) = x^2{}_f \Rightarrow Y^2{}_p(\hat{u}) = (A(f), f(\hat{u}))$. (The analysis will not be affected if π is an output partition and we restrict attention to those intelligibility functions for π that are question-independent.) Since $C^1 >> C^2$, we can then choose an x^1, $X^1{}_f(x^2{}_f)$, to strongly induce $x^2{}_f$ together with any question-independent intelligibility function of $y^2{}_p$. (Indeed, in general there can be more than one such value of x^1 that induces $x^2{}_f$.) So in particular, we can choose it so that the vector $Y^1{}_p(\hat{u}) = (A(I^2{}_{A(f)}), I^2{}_{A(f)}(\hat{u}))$ for any possible function $I^2{}_{A(f)}$. Now for that x^1, $X^2(\hat{u}) = x^2{}_f$, and therefore $A(Y^2{}_q(\hat{u})) = A(f)$, which means that $I^2{}_{A(f)}(\hat{u}) = Y^2{}_\alpha(\hat{u})$, which in turn equals $f(\hat{u})$ for that x^2. So $\forall\ \hat{u}$ such that $X^1(\hat{u}) = X^1{}_f(x^2{}_f)$, $Y^1{}_p(\hat{u}) = (A(f), f(\hat{u}))$. In other words, $X^1{}_f(x^2{}_f)$ weakly induces in C^1 the same intelligibility function of π that $x^2{}_f$ weakly induces in C^2. However since $X^1(\hat{u}) = X^1{}_f(x^2{}_f) \Rightarrow X^2{}_f(\hat{u}) = x^2{}_f$, the set of $\hat{u} \in \hat{U}$ such that $X^1(\hat{u}) = X^1{}_f(x^2{}_f)$ is $\subseteq$ the set such that $X^2(\hat{u}) = x^2{}_f$. This means that $\mathbf{l}(X^1{}_f(x^2{}_f)) \geq \mathbf{l}(x^2{}_f)$. (Our task, loosely speaking, is to bound this difference in lengths, and then to extend the analysis to simultaneously consider all such question-independent intelligibility functions f.)

Take $\{f_i\}$ to be the set of all intelligibility functions for π. By the preceding construction, π is weakly predictable to C^1 with a (not necessarily proper) subset of $\{X^1{}_{f_i}(x^2{}_{f_i})\}$ being a member of $(C^1)^{-1}(\pi)$. Now any member of $(C^1)^{-1}(\pi)$ must contain at least three disjoint elements, corresponding to intelligibility functions q with $A(Y^1{}_q(\hat{u})) = \mathbf{B}$, $\{0\}$, or $\{1\}$. (See the discussion just before Lemma 1.) Accordingly, the volume (as measured by $d\mu$) of any subset of $\{X^1{}_{f_i}(x^2{}_{f_i})\} \in (C^1)^{-1}(\pi)$ must be at least 3 times the volume of the element of $\{X^1{}_{f_i}(x^2{}_{f_i})\}$ having the smallest volume. In other words, the length of any subset of $\{X^1{}_{f_i}(x^2{}_{f_i})\} \in (C^1)^{-1}(\pi)$ must be at most -ln(3) plus the length of the longest element of $\{X^1{}_{f_i}(x^2{}_{f_i})\}$. Therefore $\mathbf{c}(\pi \mid C^1) \leq \max_{f_i} [\mathbf{l}(X^1{}_{f_i}(x^2{}_{f_i}))] - \ln(3)$.

Now take $\{x^2{}_{f_i}\}$ to be the set in $(C^2)^{-1}(\pi)$ with minimal length. $\{x^2{}_{f_i}\}$ has at most $o(2^\pi)$ disjoint elements, one for each intelligibility function of π. Using the relation $\min_i[g_i] = -\max_i [-g_i]$, this means that $\mathbf{c}(\pi \mid C^2) \geq -\ln[o(2^\pi)] + \min_{f_i} [\mathbf{l}(x^2{}_{f_i})]$. Therefore we can write $\mathbf{c}(\pi \mid C^1) - \mathbf{c}(\pi \mid C^2) \leq \ln[o(2^\pi)] - \ln(3) + \max_{f_i} [\mathbf{l}(X^1{}_{f_i}(x^2{}_{f_i}))] - \min_{f_i} [\mathbf{l}(x^2{}_{f_i})]$. The fact that for all $x^2{}_{f_i}$, $X^2(\hat{u}) = x^2{}_{f_i} \Rightarrow$

$A(Y^2_q(\hat{u})) = A(f_i) \subseteq \mathbf{B}$ completes the proof of (i).

To prove (ii), note that we can always construct one of the sets in $(C^1)^{-1}(\pi)$ by starting with the set consisting of the element of $\{X^1_{f_i}(x^2_{f_i})\}$ having the shortest length, and then successively adding other x^1 values to that set, until we get a full (weak) prediction set. Therefore $\mathbf{c}(\pi \mid C^1) \le \min_{f_i} \mathbf{l}(X^1_{f_i}(x^2_{f_i}))$. Using this bound rather than the one involving -ln(3) establishes (ii). **QED.**

Note that the set of $Z \in \mathbf{B}$ such that $[C^2]^{-1}(Z)$ exists must be non-empty, since $C^2 > \pi$. Similarly, $C^2 > \pi$ means that there is a $\hat{u}$ such that $A(Y_q(\hat{u})) = Z \subseteq \mathbf{B}$. The associated I^2_Z always exists by construction: simply define $I^2_Z(\hat{u}) = Y^2_\alpha(\hat{u})\ \forall\ \hat{u}$ such that $A(Y_q(\hat{u})) = Z$, and for all other $\hat{u}$, $I^2_Z(\hat{u}) = x$ for some $x \in Z$. Therefore the extrema in our bounds are always well-defined.

REFERENCES

[1] Adami, C. and Cerf, N.J. (2000). Physica D. **137**, pp. 62-69.

[2] Bennett, C.H. (1973). IBM Journal of Research and Development, **17**, 525-532.

[3] Bennett, C. H. (1982). International Journal of Theoretical Physics, **21**, 905.

[4] Bennett, C. H. (1987). Scientific American. **257** (11), 88.

[5] Bennett, C.H., (1987). In “Emerging Syntheses in Science”, D. Pines (Ed.), p. 297.

[6] Bennett, C. H. (1988). IBM Journal of Research and Development, **32** (no. 1), 16-24.

[7] Berger, J. (1990), International Journal of Theoretical Physics, **29**, no. 9, 985-995.

[8] da Costa, N., and Doria, F. (1991). International Journal of Theoretical Physics, **30**, 1041.

[9] Farmer, J., and Sidorowich, J. (1988). Exploiting chaos to predict the future and reduce noise, Los Alamos Laboratory report LA-UR-88-901.

[10] Feynman, R. (1986). Foundations of Physics, **16**, 507.

[11] Fredkin, E., and Toffoli, T. (1982). International Journal of Theoretical Physics, **21**, 219.

[12] Gell-Mann, M., and Lloyd, S. (1996). Complexity, **2**, pp. 44-52.

[13] Geroch, R., Hartle, J. (1986). Foundations of Physics, **16**, 533.

[14] Kanter, I. (1990). Physical Review Letters. **64**. 332.

[15] Landauer, R. (1961). IBM Journal of Research and Development, **5**, 183.

[16] Landauer, R. (1967). IEEE Spectrum. September, 105.

[17] Landauer, R., (1971). Ferroelectrics, **2**, 47.

[18] Landauer, R. (1986). Foundations of Physics, **16**, 551.

[19] Landauer, R. (1988). Nature, **335**, 779-784.

[20] Landauer, R. (1991). PHYSICS TODAY, May, 23.

[21] Lloyd, S. and Pagels, H. (1988). Annals of Physics, p. 186.

[22] Lloyd, S. (1990). Any nonlinearity suffices for computation. Cal Tech. report CALT-68-1689.

[23] Lloyd, S., and Slotine, J.-J. (1996). Int. J. Adapt. Cont. Sig. Proce, **10**, p. 499.

[24] Lloyd, S., (2000). Nature, **406**, no. 6799, p. 1047.

[25] Moore, C. (1990). Unpredictability and undecidability in dynamical systems. Physical Review Letters, **64**, pp. 2354--2357.

[26] Pour-El, M., and Richards, I. (1982). International Journal of Theoretical Physics, **21**, 553.

[27] Ruohonen, K. (1997). Complexity, **2**, no. 6, p. 41.

[28] Touchette, H., and Lloyd, S. (2000). Physical Review Letters. **84**, no. 6, pp. 1256-1259.

[29] Wheeler, J.A., and Zurek, W.H. (1983). Quantum theory and measurement. Princeton University Press, Princeton, NJ. 785-786.

[30] Wolpert, D. (1992). PHYSICS TODAY, 98, March.

[31] Wolpert, D. (1992). International Journal of Theoretical Physics, **31**, 743-785.

[32] Wolpert, D. (1990). The relationship between many-to-one mappings and thermodynamic irreversibility. Los Alamos Laboratory report LA-UR-90-4108.

[33] Wolpert, D. (1996). An Incompleteness Theorem for Predicting the Future. Santa Fe Institute working paper SFI-TR-96-03-008. See also Wolpert, D. (1991). On the computability of the

future. Los Alamos Laboratory Report LA-UR-91-3344.

[34] Wolpert, D. (1990). Complex Systems, **4**,151-200, 201-249.

[35] Wolpert, D. (1993). “The Second Law, Computation, and the Temporal (A)symmetry of Memory”, in *Advances in the Physics of Computation*, Ed. D. Matzke, IEEE press, 1993.

[36] Zurek, W. (1988). Algorithmic randomness and physical entropy. Los Alamos Laboratory report LA-UR-88-1547.

[37] Zurek, W., (1984). Nature, **341**, 119.

ACKNOWLEDGMENTS: This work was done under the auspices of the Department of Energy, the Santa Fe Institute, and the National Aeronautics and Space Administration. I would like to thank Bill Macready, Cris Moore, Paul Stolorz, Tom Kepler and Carleton Caves for interesting discussion.