Determinism

EDWARD A. LEE, University of California, Berkeley

This article is about deterministic models, what they are, why they are useful, and what their limitations are. First, the article emphasizes that determinism is a property of models, not of physical systems. Whether a model is deterministic or not depends on how one defines the inputs and behavior of the model. To define behavior, one has to define an observer. The article compares and contrasts two classes of ways to define an observer, one based on the notion of "state" and another that more flexibly defines the observables. The notion of "state" is shown to be problematic and lead to nondeterminism that is avoided when the observables are defined differently. The article examines determinism in models of the physical world. In what may surprise many readers, it shows that Newtonian physics admits nondeterminism and that quantum physics may be interpreted as a deterministic model. Moreover, it shows that both relativity and quantum physics undermine the notion of "state" and therefore require more flexible ways of defining observables. Finally, the article reviews results showing that sufficiently rich sets of deterministic models are incomplete. Specifically, nondeterminism is inescapable in any system of models rich enough to encompass Newton's laws.

CCS Concepts: • **General and reference** → **Surveys and overviews**; • **Computing methodologies** → **Concurrent computing methodologies**; • **Software and its engineering** → **Extra-functional properties**;

Additional Key Words and Phrases: Concurrency, determinism, distributed computing

ACM Reference format:
Edward A. Lee. 2021. Determinism. *ACM Trans. Embed. Comput. Syst.* 20, 5, Article 38 (May 2021), 34 pages.
https://doi.org/10.1145/3453652

1 INTRODUCTION

For most of my professional research career, I have sought more deterministic mechanisms for solving various engineering problems. My focus has always been on systems that combine the clean and neat world of computation with the messy and unpredictable physical world. Why the obsession with deterministic mechanisms? My wife, who is an expert in stochastic models, gives me a hard time about this obsession, observing, correctly, that deterministic models are just a special case. Why not, then, focus on the more general set of nondeterministic models?

Today, our society relies heavily on deterministic engineered systems. The balances in our bank accounts are a consequence of the inputs to the bank's computing systems. The email received is the email that was sent. The files on our computers contain the data that we put there. Our cars

Some of the work in this paper was supported by the National Science Foundation (NSF), award #CNS-1836601 (Reconciling Safety with the Internet) and the iCyPhy (Industrial Cyber-Physical Systems) research center, supported by Denso, Siemens, and Toyota.

Author's address: E. A. Lee, University of California, Berkeley, 545Q Cory Hall, Berkeley, CA, 94720-1770; email: eal@berkeley.edu.

1539-9087/2021/05-ART38
https://doi.org/10.1145/3453652

start when we push the start button. None of these is perfect, of course. Files and communication can get corrupted and machinery can fail. But for many systems, we easily see trillions of bits of information before encountering an error and years of reliable operation before encountering a failure.

None of this reliability comes for free. A great deal of engineering effort has gone into making the behavior of these systems repeatable. The underlying physics, a sea of electrons sloshing around, for example, offers no such repeatability. We should be in awe of these systems, but we are not because we have gotten used to their dependability. In the early days of computing, vacuum tubes would fail frequently enough that programs had to be kept short so that they would complete before a failure occurred. Computer memories were nowhere near reliable enough to replace paper ledgers. And cars frequently would not start. But today, computers operate continuously for months and years, performing billions of operations per second without error. When errors do occur, a second layer kicks in, using error-correcting codes or redundant systems to self-correct. As a result, paper ledgers have vanished. And cars start.

There are many reasons for this dependability, but for the purposes of this article, I would like to focus on my obsession: the principle of determinism. I am not talking about the philosophical view that everything that happens is an inevitable consequence of pre-existing conditions, although that concept is related. I am instead talking about an engineering principle, where a model defines exactly one correct behavior for a system in response to its inputs. Like the philosophical concept of determinism, the engineering version has many fine points, subtleties, and consequences that are subject to debate. My purpose in this article is to address these fine points. I will start informally, but I will not shy away from subtleties.

1.1 A Simple Deterministic Model

Consider a logic gate, an AND gate for example. Is it deterministic? This is, of course, a trick question. If what we mean by "an AND gate" is a piece of silicon, then the answer to this question depends on whether the underlying physics of electrons sloshing through semiconductor atoms buffeted by thermal noise under the influence of electric fields is deterministic. I will consider later in the article the question of whether modern physics can answer that question, but suffice it to say, for now, that there is a different interpretation of the question that makes it much easier to answer.

If what we mean by "an AND gate" is a model, an abstraction, that, given two Boolean inputs, produces the output "true" if the two Boolean inputs are "true" and otherwise produces the output "false," then this AND gate is deterministic. It defines exactly one correct behavior for each pattern of inputs. This model, however, leaves a great deal unsaid. When must the inputs be provided? When is the output provided? How are "true" and "false" to be represented in the physical world? Who or what is the observer of the output? What can that observer perceive about the AND gate? Determinism does not require that *all* aspects of a system be prescribed. It only requires that what we construe as "behavior" be prescribed.

Implicitly, I have assumed that what we mean by "inputs" is a pair of Boolean values from the set {*true*, *false*} and what we mean by "behavior" is producing a Boolean output chosen from the same set. This is a mathematical abstraction, not a physical artifact. If an observer is able to perceive, for example, the temperature of the AND gate, then this could be construed as part of its behavior, and the AND gate is no longer deterministic. Many possible temperatures are consistent with correct behavior of the AND gate. The abstraction also says nothing about the time at which these inputs are presented and the output is produced, so if the timing is observable, then again the AND gate is no longer deterministic. Whether something is deterministic or not, therefore, depends on what can be observed about it and whether we consider these observations part of the "behavior."

1.2 Engineering Models vs. Scientific Models

A mathematical abstraction is a model. What makes this model useful? In my book, *Plato and the Nerd* [38], I observe that the purpose of a "scientific model" is to emulate a physical system. For an "engineering model," the purpose of the physical system is to emulate the model. The mathematical abstraction of the AND gate that I have given is a rather poor model of electrons sloshing in silicon, so it is not likely to be used as a scientific model. The model is valuable because we are able to construct silicon that behaves like the model, not because it accurately describes what electrons do as they move through silicon atoms. Hence, it is more useful as an engineering model than as a scientific model.

In this article, I will focus primarily on deterministic engineering models, and hence, whether the physical world is in fact deterministic is not nearly as important as whether we can coerce the physical world to behave like a deterministic model. The reality today is that humanity has figured out how to get silicon electronics to behave like deterministic models with astonishing fidelity.

Of course, no model is perfect, and no physical realization is perfect. The famous quote from George Box, "all models are wrong, but some are useful" [6], applies to scientific models, where it is incumbent on the model to match the physical world. The mirror image, "all physical realizations are wrong, but some are useful" [43], applies to engineering models, where it is incumbent on the physical world to match the model.

In this article, determinism is a property of models, not of physical realizations. I will analyze why this property is so valuable, and why it is worth a great deal of engineering effort to build physical systems that emulate deterministic models. I will argue that using deterministic models is not at odds with dealing with uncertainty nor with building fault-tolerant systems. On the contrary, it makes these things easier. I will discuss the relationship between deterministic models and determinism in the physical world. And I will review results that show that determinism is incomplete in that sets of deterministic models that are rich enough to model the physical world do not contain their own limit points. From a practical perspective, this means that nondeterminism is unavoidable in a broad class of models.

1.3 Terminology

Determinism in philosophy is the principle that every action is a consequence of its preconditions and that fixed rules uniquely determine these consequences. The closely related principle of "causation" (or "causality") is that the preconditions and the rules *cause* the consequences. Nondeterminism, under this principle, can be thought of as uncaused action. Many people find it difficult to accept that there can be uncaused action and will go to great lengths to find some cause when none is obvious, resorting eventually to God or some other supernatural cause when all else fails.

Causation is an explanation of *why* something happens. Instead of focusing on why, we can focus on *what* happens. In this view, it is the *uniqueness* of the consequences that is the core of the concept of determinism, not causation. There is only one possible consequence of each precondition. This uniqueness is the property I focus on in this article.

Nondeterminism is closely related to the concept of randomness, but there are important distinctions. In vernacular use, a random event is an unpredictable one. But even deterministic processes can be unpredictable, so under this view, randomness does not imply nondeterminism. Nondeterministic processes are always unpredictable, however, so nondeterminism does imply randomness in this vernacular understanding of the word.

There is a more technical interpretation to the word "random," however, which seeks to quantify the *likelihood* of various possible outcomes with probabilities. Nondeterminism, in contrast, is about *possibilities*, not probabilities. It says nothing about likelihoods.

This more technical interpretation of randomness leads to a deeper split with nondeterminism, making these concepts almost orthogonal. Consider what we mean by “likelihood” and “probability.” In the classical “frequentist” interpretation of these concepts, a probability is a prediction of the proportion of repeated experiments that will turn out some way. For example, what fraction of die tosses will produce snake eyes? But no repeat of any experiment begins with the same preconditions as the previous experiment, so this frequentist interpretation has no need for nondeterminsim.

Some experiments *cannot* be repeated, and yet we consider them random and assign them probabilities. For example, what is the probability of a major earthquake in San Francisco in the next 30 years? This probability has no valid frequentist interpretation. In the “Bayesian” interpretation, a probability is a measure of how much we *know* about the outcomes, not a measure of how frequently they occur [38, Chapter 11]. Once again, randomness has no need for nondeterminism. If we know little about some deterministic system, we can consider its outcomes to be random and assign them probabilities. Whether the underlying system is nondeterministic becomes irrelevant.

1.4 The Practical Value of Determinism

Let me begin by pointing out that our engineering toolkits are full of extremely useful deterministic models. Differential equations are deterministic (with some important exceptions that I will discuss below). Most computer programming languages, if we exclude their mechanisms for concurrency, are deterministic. Synchronous digital logic, the most widely used electronic circuit design paradigm, is deterministic. Instruction set architectures are mostly deterministic. TCP/IP, the central protocol in the Internet, is deterministic in the sense that a stream of bits in yields the same stream of bits out, even though packets may be dropped (even deliberately to shape traffic) or arrive out of order. Computer memory is deterministic, even when it is built on unreliable components, as demonstrated, for example, by the RAID project (redundant arrays of inexpensive disks) [59]. Today, it is even possible to build deterministic cloud services on top of farms of unreliable servers.

All of the above are deterministic *models*. The underlying physical realization is almost certainly not deterministic, and considerable effort and cost has gone into building physical systems that are *sufficiently* faithful to these deterministic models. Perfect faithfulness is not achievable (“all physical realizations are wrong …”). But we would not be putting in all that effort if determinism were not quite valuable.

There are many reasons determinism is valuable. Here are a few:

(1) Repeatability. Every engineer tests a design while developing it. Such testing is valuable only under the assumption that if a system works once, it will work again in largely the same way. The behavior is repeatable given the same inputs. This principle is systematized in test-driven design, where collections of regression tests have well-defined correct behaviors. When a change is made to a design, the regression tests will reveal whether the change has resulted in some unexpected behavior.
(2) Consensus. When two agents come to a conclusion, it is often valuable that the conclusions be the same. Your bank and you usually agree on your bank balance. You both keep track of that balance using deterministic computations. Given the same input information, both will produce the same results.
(3) Predictability. If behaviors in response to certain inputs are predictable, then they do not need to be discovered through testing or by surprise after deployment. As I will explain below, determinism does not assure predictability, but nondeterminism assures unpredictability. Some deterministic models are predictable.

(4) Fault detection. A deterministic model gives an unambiguous definition of "correct behavior." This enables fault detection. Any behavior that deviates from the correct behavior is faulty behavior, meaning that some assumption has not been satisfied. An AND gate, for example, will not behave in conformance with the model if the temperature is too high. A **cyclic-redundancy check (CRC)** check on a value read from memory (something computed using a deterministic algorithm) reveals whether the memory has experienced a fault that caused a bit to flip. Both of these faults would be harder to detect without a deterministic model defining correct behavior.
(5) Simplicity. In a deterministic design, one input implies one behavior. In a nondeterministic design, it is easy to get an exponentially growing number of allowed behaviors. This makes comprehensive testing much more challenging. It can also compromise the ability to analyze a model using, for example, formal methods.
(6) Unsurprising behavior. Often, we want engineered systems to be boring. They should not surprise us with unexpected behaviors. When a computer programmer gets unexpected behaviors from a multithreaded program, for example, it can be extremely disruptive, costly, and difficult to fix [33].
(7) Composability. When building large systems out of smaller components, having a clear understanding of the possible behaviors of those smaller components becomes essential. Deterministic models of those components makes this far easier.

When engineers are forced to move beyond these deterministic mechanisms, building correct designs becomes far more difficult. Unfortunately, many of today's most popular programming frameworks supporting parallel and distributed computing make it quite difficult to achieve deterministic behaviors.

1.5 The Practical Value of Nondeterminism

None of what I have said implies that nondeterminism has no value. Nondeterministic models can also be valuable. Here are a few reasons:

(1) Abstraction. A nondeterministic model may provide a much simpler abstraction of a deterministic model [12]. Note that this does not contradict Property 5 above. The nondeterministic *model* is simpler, but if it is a sound abstraction, the number of *behaviors* allowed by the model cannot possibly be smaller than those of the model it abstracts. A smaller model may be easier to understand. It may also be easier to formally analyze, as long as the formal analysis does not require exhaustively exploring all possible behaviors.
(2) Uncertainty. Nondeterministic models are useful as *scientific* models of systems where our knowledge of their behavior is incomplete. For example, modeling a human operator of a car is probably not a reasonable task for deterministic modeling. Such a model of a human operator, however, is almost certainly a scientific model, not an engineering model.
(3) Deferred design decisions. For *engineering* models, nondeterminism can be a useful way to capture deferred design decisions. Deferred design decisions represent a different form of uncertainty. It is uncertainty about the model, not uncertainty about what is being modeled.
(4) Security. Many techniques for securing software systems rely on good random number generators. Pseudo-random number generators are deterministic, a weakness that opens a vulnerability. Seeding a psuedo-random number generator with the result of a nondeterministic process can help.
(5) Don't care. A model may have many acceptable behaviors in response to a given input. Serverless architectures in the Internet, for example, are useful because computations in

response to inputs can be mapped to any available server. It is irrelevant to correctness in what order and where these computations are performed.

(6) Surprising behavior. We don't always want engineered systems to be boring. An automated musical accompanist, for example, might be enriched by unpredictable behavior. Any artificial intelligence attempting to exhibit human-like behavior will also need to at least appear to be nondeterministic (see my book [39, Chapter 10] for a discussion of the connection between creativity and nondeterminism).

Notice that valid uses of determinism and nondeterminism are quite different from one another, which suggests that engineers should consciously choose between them. This is rarely easy. For example, when writing multithreaded or distributed software, almost all available languages, frameworks, and middleware are nondeterministic by default. Achieving determinism is left to the designer and is sometimes impossibly difficult [33, 47].

Probabilistic, stochastic, or random models (I will treat these words as synonyms) are also useful, but as I pointed out in Section 1.3, the concept represented by these words is largely orthogonal to nondeterminism. In computer security, for example, random numbers are usually generated using a deterministic algorithm that ensures a desirable empirical probability distribution on observations. A nondeterministic source of random numbers may, in fact, be hopelessly inadequate for computer security without some characterization of its probability distribution. In the Bayesian interpretation of probability, it is irrelevant whether the thing being observed is deterministic. What is relevant is whether the observation can be anticipated given prior knowledge.

1.6 The Cost of Determinism

Building physical systems that behave like deterministic models usually comes at a price. A synchronous digital circuit, for example, has to be clocked slowly enough to provide comfortable margins to accommodate delay variability due to manufacturing tolerances and temperature. These margins are a cost in performance. Some power-users of computers have discovered that they can often "overclock" their CPU without introducing too many errors. This may be acceptable for gaming, for example, but it would probably not be wise for a bank to overclock their CPUs.

For distributed systems, determinism comes at the cost of latency [77]. For applications where latency is a key performance metric, nondeterministic solutions may be a better choice. For example, some distributed applications choose to use UDP rather than TCP for network communication. The UDP protocol is simpler and faster than TCP, but it sacrifices the guarantee of eventual in-order delivery of messages. If occasional packet losses are acceptable for a particular application, then this may be a reasonable choice.

As with all engineering, there are tradeoffs. To evaluate these tradeoffs, it is helpful to have a deeper understanding of what determinism really is. We look at that next.

2 WHAT IS DETERMINISM?

John Earman, in his *Primer on Determinism*, states that "determinism is a doctrine about the nature of the world" and concludes that "a real understanding of determinism cannot be achieved without simultaneously constructing a comprehensive philosophy of science" [16, p. 21]. If instead of a "doctrine about the nature of the world" we view determinism as a property of models, then no such philosophy is needed. We can focus instead on the usefulness of the concept of determinism.

As a property of models, determinism is easy to define:

> A model is deterministic if given all the *inputs* that are provided to the model, the model defines exactly one possible *behavior*.

In other words, a model is deterministic if it is not possible for it to react in two or more ways to the same inputs.[1] Only one reaction is possible in the model. More precisely, only one reaction is *correct*; any other reaction is not one given by the model. In this definition, I have italicized words that must be defined within the modeling paradigm to complete the definition, specifically, "inputs" and "behavior."

For example, if the behavior of a particle is its position $x(t)$ in a Euclidean space as a function of time t, where both time and space are continuums, and if the input $F(t)$ is a force applied to the particle with mass m at each instant t, then Newton's second law,

$$F(t) = m\frac{d^2}{dt^2}x(t), \tag{1}$$

is a deterministic model (mostly, see Section 5).

The *same particle*, however, may have an equally valid nondeterministic model. For the same definitions of behavior and input, the statement "The particle accelerates in the direction of the net force" provides a nondeterministic model. It admits many more behaviors than the deterministic model. This latter model is a sound abstraction because the behavior of the deterministic model is among the behaviors of the nondeterministic one.

When considering a *physical* particle, the thing-in-itself rather than its model,[2] then Earman's hesitation comes to the foreground. A definitive answer to whether the actual particle is deterministic may never be possible. Any discussion of the determinism of a particle is necessarily a discussion of some model, not of the thing-in-itself, even if that fact goes unsaid. Some models of the particle are deterministic and some are not. I will address the physics of determinism in Section 5. But first, let us address some of the subtleties with models before we complicate the picture with the thing-in-itself.

3 BEHAVIOR, STATE, AND OBSERVATION

When determining whether a model is deterministic, we need to define "behavior." Many models tie behavior to the notion of "state." Newtonian physics, for example, does this by defining a time continuum and modeling the state of a system as positions and momentums at a shared "instant" in time. "Behavior" can then be defined as the evolution of this state in the time continuum in response to the inputs, which are forces. Modern physics complicates this simple picture (see Section 5), but the Newtonian models nevertheless remain useful.

3.1 The Notion of State

In their classic book on automata theory, Hopcroft and Ullman define "state" as follows:

> The state of the system summarizes the information concerning past inputs that is needed to determine the behavior of the system on subsequent inputs. [24, p. 13]

The state is all the information about its past that can affect its future behavior. In other words, the state of a system is the information about the past such that any additional information tells us nothing about its future behavior. This definition requires defining "behavior," but it also requires a notion of "past" and "future" as well as the boundary between these, the "present." This notion turns out to be problematic for very fundamental reasons. It is problematic in modern physics, but also in practical realizations of parallel, distributed, and cyber-physical systems [34]. These

[1]For a nice formalization of this concept, see Edwards [17].

[2]The philosopher Immanuel Kant made the distinction between the world as it is, what he called the thing-in-itself (*das Ding an sich* in German), and the phenomenal world, or the world as it appears to us.

systems have no well-defined “present” separating past and future. The notion of state, therefore, useful as it is, sits on shaky foundations.

In computing, automata theory is built around the notion of state (or equivalent notions working with sequences of symbols). Variants of automata theory use either a Newtonian time continuum or a discrete, countable model of time, both of which provide a “present” that separates past from future. Automata theories share with Newtonian physics the idea that time advances uniformly throughout the system and that there is a shared notion of an “instant” of time, a “present,” at which the system is in some state.

In automata theory, unlike Newtonian physics, even if time is a continuum, the *state* evolves in discrete jumps. At an instant, the state of the system changes from some value s to some other value s'. Again, “behavior” can be defined as the evolution of this state in time. When the system is distributed or concurrent, where it consists of multiple interacting components, possibly spread out over space, nondeterminism proves to be a useful way to model the uncertainty about the order in which state changes occur in the distinct components.

3.2 Input-Output vs. State-Trajectory Behavior

An alternative way to define behavior is to introduce the notion of “outputs,” data or symbols that some observer interprets as the behavior of the system in response to some input. Instead of state-trajectory behavior, we have input-output behavior. The relationship between input-output behavior and state-trajectory behavior is fascinating, subtle, and complex. The essential question is, what can an observer observe? The possible observations are the “output” of the system. But this is true even if we define “behavior” to be a state trajectory. In that case, we have implicitly defined an observer that can observe the state of the system at an instant in time. This definition will require us to define “an instant in time,” which, as we will see, becomes difficult for a distributed system.

Consider automata theory, where a state transition system may be endowed an explicit notion of inputs and outputs. Inputs trigger state transitions and outputs result from state transitions. A transition system evolves as a *sequence* of state transitions, and hence, both the inputs and the outputs are sequences of symbols from some alphabet. A sequence of symbols forms a “sentence,” and the set of all sentences that are possible defines a “language.” In an input-output interpretation of behavior, a deterministic model is one for which, given any input sentence, there is only one possible output sentence.

The fact that inputs and outputs are defined in automata theory to be *sequences* of symbols is important. An “observer” only ever sees sequences of symbols. These sequences cause no end of difficulties with modeling concurrent systems, where the order in which symbols occur may not be relevant or even well defined. Starting in the 1970s, Robin Milner and Tony Hoare pioneered methods for formally modeling such systems (see Winskel [73, Chapter 14] for a nice summary of these results). I will illustrate these difficulties with some key observations due to Milner.

In 1980, Milner published a series of lecture notes giving an elegant formalism that he called a **Calculus of Communicating Systems (CCS)** [54]. This formalism made it abundantly clear that looking only at the sentences of inputs and outputs is insufficient. For a simple illustration of this, consider a distributed system consisting of three components C_1, C_2, and C_3 running on three computers, illustrated in Figure 1. C_1 sends a message to each of the other two to initiate the computation. In response, the other two perform some deterministic computation and output a sequence of results (a sentence). Suppose C_2 always outputs AB and C_3 always outputs CD. Is the overall system deterministic?

As described, this system has no input, so, to be deterministic, it should have exactly one “behavior.” If by “behavior” we mean a single output sentence, as Milner does, then we are forced

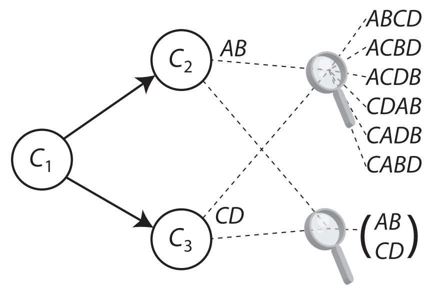

Fig. 1. Three-component system that is either deterministic or not depending on how you define the observer.

to combine the output sentences produced by C_2 and C_3. How should we combine them? If we define the "observer" to be some entity that simply observes the symbols A, B, C, and D as they are produced, then that observer will see some arbitrary interleaving of the two sentences AB and CD. There are six such interleavings, as shown at the upper right in the figure. Hence, the model is nondeterministic.

Arguably, the nondeterminism that arises here is a side effect of our insistence that an observer can only see a single output sequence. Indeed, Milner found this sort of nondeterminism unsatisfying and introduced a notion that he called "confluence." In his formalism, as long C_2 and C_3 cannot interfere with each other's ability to produce their output, this system is deemed "confluent," presented as a useful replacement for the concept of determinism.

There is another solution, however, which does not require replacing the notion of determinism. Instead, we can change what we mean by "behavior" by changing what an observer sees. If, instead of a single output sequence the behavior is defined to be a *pair* of output sequences, then the model immediately becomes deterministic, as shown at the lower right in the figure. The one and only behavior is (AB, CD), a pair of sentences, each containing a sequence of two symbols.

Redefining "behavior" may seem like a sleight of hand, a trick. It is not. *Any* definition of behavior depends on a notion of an "observer," and fundamentally, for any system, different observers see different things. Let me make this crystal clear with a trivial example. Consider the following C program:

```
int main(int argc, char* argv[]) {
    printf("Hello World.\n");
}
```

In this program deterministic? If the "observer" is a human sitting at a computer screen, then for this program to be deterministic, that observer should be able to see only exactly one possible observation from running this program. Is that the case? What color will the characters "Hello World" be rendered in? How long will it take before the observer sees "Hello World"? Neither of these observable properties are specified by the program, so if we include these properties in the notion of "behavior," then the program is nondeterministic. Almost certainly, however, this is not what we intended. If we carefully define "observer" and restrict that observer to observing the sequence of symbols produced by the program, then the program becomes deterministic.

3.3 Alternative Conceptions of Behavior

Returning to our example with C_1, C_2, and C_3, a different notion of deterministic programs was introduced by Gilles Kahn in 1974, a class of models that are now called **Kahn Process Networks (KPNs)** [26]. These are closely related to dataflow models [40, 41], which share with them a different notion of "observer" from Milner's that leads to different conclusions about determinism. A KPN is a network of processes that send messages to each other along defined channels, where each channel is assumed to preserve the order of the messages and deliver them reliably (like TCP). Kahn gave an elegant construction using the mathematics of partial orders to give conditions on the processes such that the overall behavior of the program is deterministic [26]. Kahn and MacQueen later showed that a simple constraint on the processes in the network, "blocking reads," is sufficient to ensure determinism [27].

The definition of "behavior" here, however, is not the sentences of Milner. Kahn defined behavior to be a collection of (possibly infinite) sequences of messages, each recording the messages traversed on one channel. "Behavior" in a KPN, therefore, is a tuple of possibly infinite sequences. Under the KPN model, C_1, C_2, and C_3 together form a deterministic system. No notion of confluence is needed.

Kahn networks also do not require any notion of state. Each process can be usefully modeled as a state machine, but there is no need to ever talk about a global state, some combination of the states of all the processes at some instant in time. The approach Kahn took toward defining the semantics of programs without appealing to a notion of state can be generalized to many other kinds of concurrent systems [32], as done, for example, in the tagged signal model [42].

An issue with Kahn networks is that coordinating decision making in a distributed system can become difficult [47]. There is no notion of "the state of the system," so writing code like "if the state of the system is X do Y" becomes challenging.

A number of alternative concurrency models have emerged that preserve determinism and reintroduce a semantic notion of state without having to appeal to confluence. One such alternative concurrency model is embodied in synchronous-reactive languages [5]. These languages introduce the notion of a global "clock" that ticks discretely, much like that found in synchronous digital logic design. At each tick of this now conceptual (rather than physical) clock, many computations are performed, possibly in parallel, until the entire system settles to a well-defined state. With certain constraints on the component computations and on their scheduling [19], the resulting state is a unique function of the inputs. The resulting model can be conceptualized as lying somewhere between Milner's transition systems and Kahn's asynchronous process networks. During the computation, at a tick, the behavior is more like a Kahn network, where data precedences constrain the order in which things happen, but no notion of a single global state trajectory is needed and computations can proceed in parallel and asynchronously. At the conclusion of a tick, things settle to a well-defined state, enabling a higher-level state-trajectory model that treats all the computation at a tick as a single atomic state transition. These models are deterministic as long as the observer is constrained to observe the state only at the conclusion of each tick. The execution of synchronous-reactive languages may be thought of as "punctuated chaos," where periods of chaotic, parallel, asynchronous computation are marked by isolated points of stable, well-defined state.

3.4 Time

Although synchronous-reactive languages have "clocks," they do not really have a notion of time. For **cyber-physical systems (CPSs)**, which combine computation and networking with physical components [34], some notion of time becomes essential [35]. It is not necessary (nor is it physically possible) to insist on the Newtonian notion, where time is a continuum with well-defined instants t that are shared by all components in the system. Instead, leveraging the punctuated chaos of synchronous-reactive languages, we can assign a semantic measure of time elapsed between ticks [68, 72]. This makes it possible for models to combine Newtonian models of physical components with computational models, thereby offering a rigorous approach to CPS design that does not sacrifice determinism [13].

We can make an even bigger commitment to a notion of time by explicitly timestamping events, as done in discrete-event systems [31]. In such models, the concept of a global "tick" is replaced by timestamped messages with the constraint that every component processes messages in timestamp order. With an additional constraint that messages with identical timestamps be processed in a well-defined order, the model becomes deterministic. This is the principle behind the recently introduced reactor model [46] as realized in the Lingua Franca language [48]. This language is the current manifestation of my obsession with determinism. Semantically, such discrete-event models can be viewed as a generalization of synchronous-reactive models [44] and have even been called "sparse synchronous" models by Edwards and Hui [18].

3.5 Observers

Another major issue that emerges from Milner's calculus is the tension between state transitions and output sentences. Which are observed? It is easy to construct an automaton that makes nondeterministic state transitions and yet produces a deterministic output in response to inputs. Should this automaton be deemed deterministic or not? This tension has prompted some researchers to attempt to codify this distinction, sometimes using two distinct words, "determinate" and "deterministic" (see, e.g., von Hanxleden et al. [71]). The word "determinate" is meant to capture the idea that observable outputs are uniquely defined by the inputs, whereas the word "deterministic" attempts to capture the idea that there is a unique way in which the outputs are determined. Using this distinction, our constructed automaton (the one that makes nondeterministic state transitions and yet produces a deterministic output) is determinate but not deterministic.[3]

However, this distinction is specious. An automaton is a model, not an implementation. An implementation might be realized by a computer program, in which case, the automaton is a model of the behavior of that program. The program itself is a model of the computations to be performed by one or more **instruction set architectures (ISAs)** communicating over some network fabric using some protocols. The ISA and the network protocols are themselves models of a physical system with electrons sloshing around. At which level should we determine whether the system is deterministic? How many words for determinism will we need to cover all these levels of models? If we are clear about what we mean by an "observer," then no such distinction is needed and one word is sufficient. Determinism becomes a property of the combination of the model and the observer.

Another context in which this tension comes to the foreground is with Alonzo Church's lambda calculus [8], a model of computation in which "lambda expressions" are subjected to syntactic rewriting following a set of rules. The Church-Rosser theorem shows that if such an expression can be reduced to the point where none of the rewriting rules can be applied anymore, then the same final expression results regardless of the order in which the rewriting rules are applied [9]. This model of computation can be construed as nondeterministic if the observer can see the intermediate expressions or as deterministic if only the final expression is visible.

The notion of an observer, it turns out, has its own subtleties. Should an observer be passive and objective, or can the observer interact with the system? This distinction turns out to be important.

[3] A similar distinction is given by Wisniewski, et al. [74], who define "strong" and "weak" determinism in terms of Petri nets augmented with inputs and outputs. They make a distinction between "stable markings" (ones where no transitions are enabled, given the inputs) and "unstable markings" (which can be viewed as transitory markings toward stable markings). A "weakly deterministic" Petri net is one where for each possible stable marking and input, there is exactly one successor stable marking, and a "strongly deterministic" Petri net is one where for any marking and input, there is exactly one successor marking. Another similar distinction is given by Khomenko et al. [28], who define "output-determinacy" as a relaxation of determinacy.

In computer security, for example, any discussion of the security of the system requires a threat model that is explicit about the capabilities of an attacker. It is important whether the attacker can interact with the system or is restricted to passively observing it.

Milner's own definition of determinism exposes such subtleties in the concept of an "observer." Milner's definition of determinism depends on a relation between automata that Milner called "bisimulation." It is beyond the scope of this article to explain bisimulation, but suffice it to say that this concept depends on an observer that is not just passive and objective, but rather can interact with the system being observed.[4] In 1980, David Park found a gap in Milner's prior and simpler notion of "simulation," in which a passive and objective observer automaton emulates the behavior of an observed automaton [58]. Park noticed that even if two automata simulate each other, they can exhibit significant differences in behavior. These differences are not observable by any passive, objective observer, but if the observer can interact with the observed automaton, providing inputs that depend on its observations, then the differences become visible. Park's observation led Milner to develop the notion of "bisimulation" (and the closely related notion of "observational equivalence"), an interactive form of simulation that ensures that two automata are indistinguishable even through interaction. He then based his notion of determinism on bisimulation [55].[5]

4 DETERMINISM VS. PREDICTABILITY

Determinism does not imply predictability. For a model to be predictable, we must be able to anticipate its behavior by examining the model rather than by just watching what it does. Once again, we must be careful to define "behavior." No computer program is predictable, for example, if "behavior" includes generating heat. The computer program alone is not sufficient to anticipate how much heat will be generated by executing the program.

Turing machines are deterministic models. The "input" to a Turing machine is a binary bit sequence, which Turing described as the initial sequence of marks on a tape. The "behavior" can be defined to be a final bit sequence, the final sequence of marks on the tape when the machine halts, or a special result, often called "bottom" and written with the symbol $\perp$, that indicates failure to halt. A Turing machine, therefore, is a model whose operation computes a function that maps an input bit sequence to either an output bit sequence or $\perp$.

A Turing machine is also deterministic to an observer that can observe the sequence of operations that lead to this final output, but this stronger form of determinism is less important to the notion of "computation." In fact, one of the most interesting things about the foundational theory of computation is that many different mechanisms, including some nondeterministic ones, can be used to compute the same set of functions. A human using pencil and paper and following well-defined rules, given enough paper and time, can compute the same set of functions. Your laptop, which uses low-level operations that are quite different from those of a Turing machine and involve no tape, given enough time and memory, can also compute the same set of functions. Church's lambda calculus, which has expressiveness equivalent to Turing machines, is deterministic in the weaker sense, where "behavior" is the final irreducible expression, but its mechanisms for finding that expression need not be deterministic. Reduction rules may be applied in any order. The low-level mechanisms are not as important as the function that is computed, so we will stick with the definition of "behavior" that restricts the observer to observe the final binary result.

[4]See my book [39, Chapter 12] for an in-depth discussion of the distinction between observation and interaction. That chapter includes a gentle introduction to the concept of bisimulation.

[5]Sangiorgi [67] gives a nice overview of the historical development of this idea. He notes that essentially the same concept of bisimulation had also been developed in the fields of philosophical logic and set theory.

The question now becomes, is the deterministic behavior of a Turing machine predictable? Alan Turing showed that it is not. He showed that there is no mechanism, no systematic procedure that can predict, for all Turing machines, whether an execution will halt for a particular input [70]. This result carries over to all the similarly expressive mechanisms, including Church's lambda calculus, a human with paper and pencil, and your laptop. This result decisively decouples determinism from predictability. Determinism does not imply predictability.

Turing machines and lambda calculus underlie modern imperative and functional programming languages, respectively. Programmers, therefore, face the possibility that they will not be able to predict whether their programs will terminate. Usually, however, a programmer needs assurances that the program will terminate (e.g., a program that converts a text file into a PDF file) or will not terminate (e.g., a web server or an operating system). Fortunately, most programs are predictable in this sense, even if, in theory, it is impossible for all programs to be predictable in this sense.

Many computer programs are unpredictable in a more informal sense. In this more informal sense, the question is, by examining the program but not executing it, can we anticipate its behavior? Some programs are designed to be unpredictable in exactly this sense. Pseudo-random number generators, for example, fall in this category. You cannot tell from looking at the program what numbers it will generate. You must execute the program instead. Machine-learning algorithms turn out also to be unpredictable in this sense. Cellular automata, a class of deterministic computational machines that are also unpredictable in this sense, are capable of surprising and complex behaviors, so complex as to prompt some thinkers to conclude that they underlie all of physics [75].

Unpredictable deterministic models also arise in Newtonian physics. Nonlinear differential equation models, such as those modeling the thermodynamics of weather, exhibit behaviors that are so unpredictable that they are called "chaotic" [49]. A chaotic model is one where arbitrarily small perturbations in the inputs or the initial conditions have arbitrarily large effects in the future.

Deterministic chaotic systems are, *not even in principle*, predictable. Lewis and MacGregor, in 2006, proposed a thought experiment involving two spheres colliding with each other within a contained space [45]. They calculate the precision with which the initial conditions must be known in order to predict the behavior after a certain number of collisions. They then assume that the initial positions of the spheres are to be determined optically and derive the wavelength of light that will achieve the required precision. They then show that a single photon of such light would have "more energy than is currently posited for the entire universe in order to resolve the initial state of the system with precision sufficient to predict its behavior after just 35 collisions" [45, p. 10–11].

The "system" that Lewis and MacGregor analyze is, however, a model, not a physical system. The question we should be asking, therefore, is whether the *model* is predictable. The model, given by Newton's laws with discrete collisions, admits no closed-form solution, and therefore would have to be solved numerically to predict its behavior. A similar analysis could be done to determine the arithmetic precision and computational load required to accurately predict the behavior after 35 collisions. I have not done this analysis, but I suspect it would show that this is equally impossible.

4.1 Murphy's Law and Faults

An argument that I hear frequently against deterministic models goes something like this: In the real world, things will go wrong. Nothing is really predictable. Even deterministic models will be violated in practice, so why bother with deterministic models? Why not, instead, assume everything is random and design your system to tolerate this randomness?

This argument, if carried too far, suggests we should not bother with the reliable in-order packet delivery of TCP, the linchpin of the Internet. We should not bother with CRC checks in computer memories. We should not bother with synchronous digital logic design, and instead use circuits that may or may not produce expected results. But we do bother with all these things. Why?

Today, computer memories have replaced ledgers in finance and law. Digital signatures have become an acceptable way to finalize legal contracts. Stock trades execute without human intervention or paper records. None of these would be possible without deterministic models and our ability to build physical systems that are highly faithful to these models. Electronic circuits perform billions of arithmetic operations per second and go for years without errors.

I repeat the core principles because they are so important. Determinism is a property of models, not of physical systems. An *engineering* model is a specification of how a physical system *should* behave, not a model of how the physical system *does* behave (the latter is a scientific model). When you have a model that defines how a system *should* behave, then you get, for free, the notion of a "fault." A fault is a behavior that deviates from the specification.

The existence of faults does not undermine the value of deterministic models. In fact, the very notion of a fault is strengthened by deterministic models because they define more clearly what behavior a physical system *should* have. Detecting faults, therefore, is easier. A CRC, for example, detects at least some violations of a simple deterministic model of a computer memory. This enables fault-tolerant design, where the system reacts in predictable ways to faults.

Every realization of an engineering model can exhibit faults. When we successfully build a physical system that reliably behaves like a model, it does so only under certain assumptions. No computer will correctly execute a program if it overheats, is crushed, or is submerged in salt water. The model is faithfully emulated only under the assumption that none of these things has happened.

Making the assumptions clear also has value. The Ptides model [77] for deterministic distributed execution of discrete-event programs, for example, which is realized in Google Spanner [11] and Lingua Franca [48], assumes a bound on clock synchronization error and a bound on network latency to achieve extremely efficient distributed and fault-tolerant coordination of program components. Violations of these assumptions will occur in practice, but in a well-designed system, they will be rare. Moreover, such violations are detectable because they manifest as software components seeing events out of timestamp order. A deterministic model, together with clearly stated and quantified assumptions under which a physical realization emulates the model, enables efficient designs that can react in predictable ways to faults. Hence, despite Murphy's Law, deterministic models are useful, even in the face of unpredictable failures.

5 DETERMINISM IN PHYSICS

The question of whether the physical world is deterministic has been controversial for a long time. In the early 1800s, Pierre-Simon Laplace argued that if someone (a "great intellect," later known as "Laplace's demon") were to know the precise location and velocity of every particle in the universe, then the past and future locations and velocities for each particle would be completely determined and could be calculated from the laws of classical mechanics [30]. Is this true?

In 2008, David Wolpert proved that Laplace's demon cannot exist [76]. No such calculation is possible. Wolpert's proof relies on the observation that such a demon, were it to exist, would have to exist in the very physical world that it predicts. This results in a self-referentiality that yields contradictions, not unlike Turing's undecidability and Gödel's incompleteness theorems.

But Laplace's demon is about *prediction*, not just determinism. We already know that determinism does not imply predictability. Even if prediction is known to be impossible, we cannot conclude that the world is nondeterministic. We will look at the question of whether Newtonian physics, the state of the art when Laplace lived, is a deterministic model. You may be surprised by the answer.

More recently than Laplace, Karl Popper, high priest of scientific positivism, also insists on a deterministic universe:

> One sometimes hears it said that the movements of the planets obey strict laws, whilst the fall of a die is fortuitous, or subject to chance. In my view the difference lies in the fact that we have so far been able to predict the movement of the planets successfully, but not the individual results of throwing dice. In order to deduce predictions one needs laws and initial conditions; if no suitable laws are available or if the initial conditions cannot be ascertained, the scientific way of predicting breaks down. In throwing dice, what we lack is, clearly, sufficient knowledge of initial conditions. With sufficiently precise measurements of initial conditions it would be possible to make predictions in this case also. [61, p. 198]

This quotation reflects a conventional wisdom, which dictates that Newton's laws provide a deterministic model of the universe. Also conventional wisdom is that quantum physics dashed that determinism. Neither of these is strictly true. A concise summary of the ways that determinism in these physics models have been interpreted is given by Hoefer [23]. A more in-depth study is given by Earman's *Primer on Determinism* [16]. Here, I will relate some of these interpretations to the above discussion of determinism in engineering models and give my own perspective on the subject.

5.1 Nondeterminism in Newtonian Physics

A (rather controversial) example of nondeterminism in Newton's laws is due to the philosopher of science John Norton [57].[6] Norton considers a point mass precariously balanced on top of a smooth frictionless dome. Norton shows that, without violating any of Newton's laws, the mass can spontaneously begin sliding down the side of the dome in an arbitrary direction at an arbitrary time without anything causing it to start sliding. His argument is carefully constructed and surprised me when I first heard it. I was sure his argument was wrong, but I finally concluded that my certainty was based on circular reasoning.

Newton's second law, given in Equation (1), states that at any time instant, the force imposed on an object equals its mass times its acceleration. If there is no force, the acceleration must be zero. If the acceleration is zero, then the velocity is not changing. Hence, it would seem that if the mass is not moving, balanced on the top of the dome, and no force is applied, then it should remain still, with velocity equal to zero. But Norton points out that it is possible for the mass to start sliding down the dome at any arbitrary time T without violating this law and without any force initiating the slide. At the instant T, the mass will have velocity zero and acceleration zero, so it is not moving. But at any time greater than T, say at $T + \epsilon$, no matter how small ϵ is, the mass may be no longer centered on the top of the dome. It will now be sitting on a slope, which means that gravity will exert a nonzero force in the downhill direction, and the mass will have a nonzero acceleration.

Specifically, Norton proposes a dome shape where the dome drops by a distance $h = (2/3g)r^{3/2}$, where r is the distance along the surface of the dome from the center of the dome and g is the force of gravity (see Figure 2).[7] There is nothing particularly special about this shape; it just makes the math work out simply. With this choice, the force on the mass tangent to the dome, as a function of the distance r from the center of the dome, is $F(r) = \sqrt{r}$. With this function, Equation (1) admits many solutions. In particular, if we assume unit mass $m = 1$, then the following is a solution for

[6] Another example with similar properties is given by Dhar [14].

[7] Gareth Davies points out in his blog that this is a rather odd specification of the dome shape and that the extent of the dome has to be limited to $r \leq g^2$. See https://blog.gruffdavies.com/2017/12/24/newtonian-physics-is-deterministic-sorry-norton/.

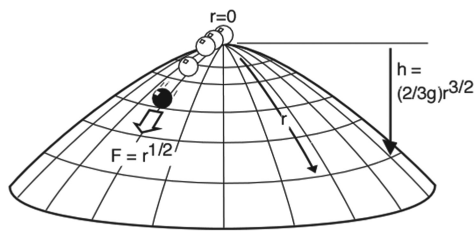

Fig. 2. Norton's dome (from http://www.pitt.edu/~jdnorton/Goodies/Dome/index.html).

any T:

$$r(t) = \begin{cases} (1/144)(t-T)^4 & t \geq T \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

At the instant $t = T$, the mass is not moving, the net force on the mass is zero, and the mass is not accelerating. At any time larger than T, the mass is moving, the net force is not zero, and the mass is accelerating down the dome. T can have any value without violating Newton's second law. Equation (1) holds at every instant t.

There are many subtleties around Norton's example. First, it may be helpful to realize that there is no *first* instant at which the mass accelerates. Instead, the time $t = T$ is the *last* instant at which the mass is *not* accelerating. At all instants greater than T, there is a nonzero net downward force, gravity on a slope, so the mass accelerates. Zinkernagel [78] claims that this property is the key flaw that would allow us to declare Norton's dome to not be a "Newtonian system." He says the force lacks a "first cause." But requiring a first cause would force us to reject many other innocuous systems that do not exhibit nondeterminism, including many reasonable models where force appears gradually [20].

Malament [50] offers a fascinating analysis of the mathematics behind Norton's dome. First he addresses a preconception that was part of what made me initially skeptical about Norton's claim. I had always assumed that Newton's second law (Equation (1)) would have a unique solution for any input force function F that could be generated by a reasonable Newtonian system. Norton's system seems reasonable in this sense because the force is just Newtonian gravity and the dome is a reasonably simple geometric shape. But Malament points out that uniqueness is not guaranteed if F is not continuously differentiable. Norton's function is not continuously differentiable at $r = 0$, and Equation (1) admits many solutions. We could restrict "Newtonian Systems" to include only forces that are continuously differentiable.[8] This is not satisfactory to me, however. We would have to rule out a large number of shapes, including shapes that do not lead to nondeterminism, such as a table with an edge where a mass slides off the edge.

Another possible objection to Norton's dome is that such a shape could never be constructed perfectly. This argument is specious, however, because *every* shape that we can describe mathematically will have flaws when constructed in the physical world. This objection would effectively eviscerate Newtonian physics.

Yet another approach to rejecting Norton's claim is to impose constraints on the *solutions* to Equation (1) rather than force function. For example, Davies suggests that higher-order derivatives of the solution need to exist and be continuous for the solution to be a reasonable model of physical behavior.[9] The family of solutions given by Equation (2) does not meet this requirement. However, this too is not satisfactory to me. Again, it would rule out a mass sliding off the edge of a table.

[8] A weaker but sufficient constraint would be to require the force to be locally Lipschitz.
[9] https://blog.gruffdavies.com/2017/12/24/newtonian-physics-is-deterministic-sorry-norton/.

Moreover, this amounts to saying something like "the model is valid if we choose from among its behaviors the one that seems reasonable."

It seems that to regain determinism, we need to augment Newtonian physics with additional axioms that are not derivable from the core concepts. Newton could have given us a fourth law of motion going something like this: "a mass can only have one possible motion that conforms with the previous three laws." This would assume away nondeterminism in order to obtain a deterministic model of the physical world.

Fletcher [20] points out, however, that even this restriction has several possible versions, and any choice between these seems arbitrary. For example, we could declare Norton's dome to not be a "Newtonian system" altogether, or we could declare it to not be a "Newtonian system" only when the mass starts or ends at the peak of the dome. If the initial position of the mass is somewhere else on the dome, Newton's second law gives us a deterministic model. What if it starts somewhere else on the dome with some momentum toward the peak and crosses the peak? At the moment of crossing, does the system suddenly and instantaneously cease to be a Newtonian system?

Are Newton's other laws violated when the mass spontaneously starts sliding? Newton's first law states that an object will remain at rest or in uniform motion in a straight line unless acted upon by an external force. It may seem that having the mass spontaneously start to move violates this law, but actually it does not, at least under one reasonable interpretation of this law. Since an external force may vary in time, this first law needs also to be interpreted as a statement that holds at each instant of time. Under this interpretation, the first law becomes a special case of the second law where $F(t) = 0$.

There is another interpretation of Newton's first law, however, that restores determinism for this example, but this interpretation requires more than Newtonian physics and is ultimately based on circular reasoning. The first law can be interpreted to mean that there can be no *uncaused* changes in momentum, where the cause is some external effect. If the mass slides off the dome as I have described, then there must be some such external effect causing this. That effect must be something other than a Newtonian force, however, because any force would lead to a trajectory different from Equation (2), which satisfies the second law. So what could that required external non-force effect be? The force of will of a conscious mind? The force of God? There is nothing in Newtonian physics that qualifies.

One could take this absence of a model for such an external non-force as an argument that the mass will not move. But the *absence* of a model cannot be construed as evidence. According to Norton, there is no *need* for such an external non-force for the mass to slide down the side of the dome, so there is no need for a model for this external non-force. Newton's laws are still satisfied. It is equally valid to demand a model for whatever keeps the mass perched on the dome. What non-force is that?

It will still disturb many readers that the mass can start moving with no provocation. Newton's laws are *also* satisfied by a mass that behaves itself and remains quietly perched at the top of the dome for all eternity. Isn't it reasonable to assume it will do that? An empirical approach would actually find this interpretation *unreasonable*, since any practical realization of Norton's dome will result in the mass sliding off the peak. The apparent reasonableness of this interpretation is due to a distaste for uncaused action, i.e., a distaste for nondeterminism. It is not due to empirical evidence and indeed flies in the face of empirical evidence. Hence, the argument that the mass will remain at the top of the dome is circular. It is not supported by the mathematics of Newton's laws alone and instead depends on the assumption that nothing happens without provocation. In other words, it concludes determinism based on an assumption of determinism.

The presupposition that every behavior has a cause is a difficult one to give up. In 1913, Bertrand Russell challenged the scientific world to give it up:

> All philosophers, of every school, imagine that causation is one of the fundamental axioms or postulates of science, yet, oddly enough, in advanced sciences such as gravitational astronomy, the word “cause” never occurs. … The law of causality, I believe, like much that passes muster among philosophers, is a relic of a bygone age, surviving, like the monarchy, only because it is erroneously supposed to do no harm. [66]

Objective discussion of causality is difficult because the notion of causality lurks in every aspect of natural language.[10] In my recent book, *The Coevolution* [39], I examine this presupposition of causality in much more depth, leveraging the arguments of Judea Pearl in [60] (who argues that reasoning about causality requires subjective involvement) and evolutionary biologists (who argue that the notion of causality may have arisen because of its evolutionary survival value rather than because it is a fact about the world). But for our purposes here, it is sufficient to simply observe that Newton’s laws do not imply causality.

What about Newton’s third law, which states that every action has an equal and opposite reaction? When gravity exerts a force on a mass, causing the mass to fall, the mass exerts an equal and opposite force on the earth, causing the earth to rise. Since the mass also gains momentum in some lateral direction, the earth must acquire an equal and opposite lateral momentum. The mass of the earth, however, is so much larger than the masses in Norton’s example that, to the earth, the force exerted on it is negligible. At all times $t > T$, the nonzero net force downward and laterally on the mass will be balanced by an equal and opposite nonzero net force pulling the earth up and laterally. The effect of that force will not be measurable, but it is there, so the third law is also not violated.

Newtonian physics is often assumed to be time reversible, but reversing time on Norton’s example has curious effects. Consider the scenario where the mass starts on the outskirts of the dome and we push it with just enough force that it reaches the top of the dome and stops. If we push too hard, it will go over the top of the dome. If we don’t push hard enough, it will not reach the top of the dome and will fall back down. But if we push it with the Goldilocks force, just right, it will stop at the top, stay there for an arbitrary amount of time, and then spontaneously slide down the dome again sometime in the future. As soon as the mass perches at the top, it’s history is lost. Nothing in its state reveals when the mass arrived at the top, thereby foiling Laplace s demon.

For this scenario to work, Norton points out that the shape of the dome is important. If the dome is a perfect hemisphere, then with the Goldilocks force, it will take infinite time for the mass to reach the top of the dome. It will keep slowing down as it approaches the top, but it will never actually reach the top. But there are many other dome shapes where the mass reaches the top in finite time. Norton’s example, where the dome drops by a distance $h = (2/3g)r^{3/2}$, is one such dome shape.

In conclusion, either Newtonian physics admits nondeterminism or Newtonian physics needs to be augmented with additional axioms that preclude nondeterminism. As I will show in Section 6, however, any set of additional axioms that preclude nondeterminism will also preclude many useful models of physical phenomena.

5.2 Metastable States

It turns out that many systems are vulnerable to similarly uncaused action under Newtonian physics. When the mass is perched on the top of the dome, it is in a metastable state. A metastable

[10] A nice collection of essays on the deep influences of the notion of causality on language is found in Copley and Martin [10].

state is marginally stable, where infinitesimal disruptions throw the system out of its precarious state. Norton's mass is vulnerable to falling out of its metastable state with no provocation.

Electronic circuits, particularly ones at the boundary between the continuous physical world and the discrete world of digital electronics, have long been known to be vulnerable to lingering for unbounded periods of time in a metastable state [29, 51, 53]. So the problem is not limited to cute examples of masses on domes. It is a fundamental problem at the boundary between the discrete, computational world of computers and the continuous physical world.

A particular kind of metastable system is a bistable system, one that has exactly two stable states and can persist for an indeterminate period of time in a metastable state between the two stable states. A digital circuit can be thought of as a piece of electronics that wants to be in one of two states, and in principle, situations where it can linger indefinitely between these two states are unavoidable.

Designers of such circuits go to great lengths to make sure that the probability of lingering gets extremely small as time advances. Consequently, it is rare for circuits to persist in a metastable state for very long. Such situations can occur, however, but are difficult to reproduce in the lab. They have occasionally been implicated in otherwise inexplicable crashes of computers. Is the basic operation of such circuits deterministic? If not, then any electronic implementation of a Turing machine, a deterministic model, is actually nondeterministic.

Bistable behavior has also been observed by biologists in nerve axons. Under certain circumstances, these axons can linger for an indefinite period of time before settling into one of two resting potentials [15]. It is likely that such metastability plays a role in brain function.

Building a physical realization of Norton's dome is impossible because no physical dome is perfectly smooth and frictionless and no mass is a point mass. Nevertheless, it is common to deliberately build systems that come as close as possible to such metastability. Sensitive instruments depend on metastable states. The instrument hovers in its (nearly) metastable state until the slightest nudge from the thing being measured pushes it off in one direction or the other. The circuit that reads the contents of a DRAM, a commonly used computer memory, for example, makes use of such metastability to read a tiny stored charge.

5.3 Relativity

The special and general theories of relativity are mostly deterministic in a similar sense that Newtonian physics is mostly deterministic. There are only a few corner cases, specifically singularities, that result in many possible futures given a specific past. The event horizon of black holes presents such singularities, but these are unobservable in the rest of space time, and hence arguably pose no problem to the determinism of the theory. A conceivable class of singularities, called "naked singularities," however, have no event horizon and become observable. In 1969, Sir Roger Penrose posited that such naked singularities do not exist in the universe, a principle called "cosmic censorship." If this principle holds, then general relativity adds no sources of nondeterminism over and above any already present in classical physics.

Nevertheless, there is one aspect in which relativity complicates the notion of a deterministic model of physics. Specifically, it undermines the notion of "the state of the system," as used by Milner and Newton, and consequently, we can no longer talk about determinism in terms of the evolution of the state of the system in time. In relativity, there is no "instant in time." This Newtonian concept is replaced by a Cauchy surface, a hypersurface in four-dimensional space time. But the Cauchy surface is different for each observer, so two distinct observers can disagree about the state of the system.

The difficulty can be illustrated with a simple example, illustrated in Figure 3. Consider a distributed system with two physically separated subsystems, one of which makes an instantaneous

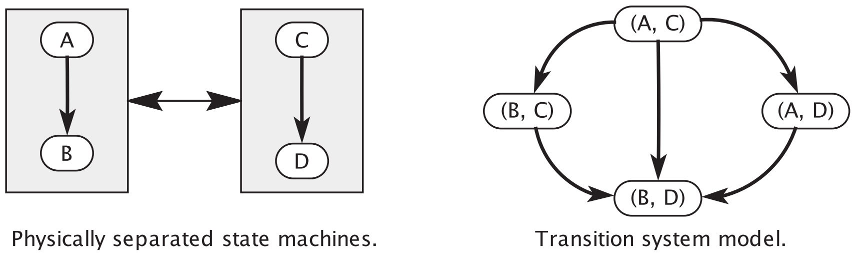

Fig. 3. Model of a distributed system that relies on a notion of "system state."

transition from state A to B, and the other of which makes a transition from C to D. Under a Newtonian model of time, one of the following transition sequences must be the true one, in the sense that it is actually what happens when the system transitions from (A, C) to (B, D):

$$\begin{aligned}&(A,C) \to (B,C) \to (B,D)\\&(A,C) \to (A,D) \to (B,D)\\&(A,C) \to (B,D)\end{aligned}$$

However, according to relativity, it is possible for none of these to be "true" in all frames of reference. The order of physically separated events may depend on the observer. It is not that two different observers just *see* things delayed in different ways; it is that the true order for the different observers is different. There is no ground truth, in the sense that we cannot make a statement like "at time t, the system was in state (B, C)." That statement can be true for one observer and false for another.

This problematic notion of time (and hence state) follows easily from the fact that the speed of light is the same for all observers. Consider a simple thought experiment, a variant of the famous Einstein Train example. Suppose that Randy is seated at the center of a rapidly moving train car, and Jane is standing on a platform at a station where the train does not stop. Suppose that just at the instant that Randy passes Jane, he hits a button that emits a brief pulse of bright light. In Randy's frame of reference, the light will hit the front and back ends of the train car at the same time. However, in Jane's frame of reference, the light will hit the back of the train car before it hits the front. In Randy's frame of reference, the light has to travel the same distance in either direction, and since the speed of light is constant, it will strike both ends of the car simultaneously. But in Jane's frame of reference, the distance the light has to travel to the back of the train is less because the back of the train is moving toward where the light was emitted while the light is traveling toward the back of the train. Hence, the two events of light striking the ends of the train are simultaneous for Randy but not for Jane.

Note this example is not just a cute toy that is only realizable if you have trains that can travel close to the speed of light. The phenomenon is called the Sagnac effect, and it is used in ring laser gyroscopes, a key component of many inertial guidance systems, including many used in commercial aircraft. The Sagnac effect also has to be taken into account in the design of GPS because of the rotation of the earth.

Instead of an evolving state in time, a relativistic system is a collection of events where, given a pair of events, one may precede the other. This is analogous to the contrast between Milner's transition systems (analogous to classical physics) and Kahn's precedence order constraints (analogous to relativity). Of course, if one event *causes* another, then relativity is careful to ensure that all observers see the one before the other, so nothing magical happens. To ensure this consistency, relativity posits that the effects of any event cannot propagate through space faster than the speed of light.

Interestingly, a similar dichotomy exists in the philosophical study of time, dubbed an "A-Series" (classical) model of time or a "B-Series" model by Gale [21]. An A-Series model of time is built on the tensed notions of past, present, and future, while a B-Series model of time is built on a partial order, a "precedes" relation between events. For example, the statement "I will complete this paper today" is an A-Series statement and will have a different meaning if uttered tomorrow. In contrast, "Completion of this paper precedes the 2021 New Year" is a tenseless B-Series statement. It has the same meaning whenever it is uttered. Relativity requires a B-Series model of time.

The illusion of state is a powerful illusion, however, because no single observer can observe a system to be in two states at once. For example, no observer of the example of Figure 3 will see both (B, C) and (A, D). This becomes important when considering quantum models, which I do next.

5.4 Quantum Physics

It is common to assume that quantum physics immediately undermines any notion of the world being deterministic, but the story is more subtle. There are many interpretations, not all of which lead to nondeterminism.

First, under quantum mechanics, the evolution of the particle's wave function is deterministic, following the Schrödinger equation. If we redefine "behavior" to be the evolution of the wave function in time, rather than position and momentum, then the model is deterministic. The fact that the Schrödinger equation is deterministic prompted Stephen Hawking to colorfully proclaim that determinism can be salvaged:

> At first, it seemed that these hopes for a complete determinism would be dashed by the discovery early in the 20th century that events like the decay of radioactive atoms seemed to take place at random. It was as if God was playing dice, in Einstein's phrase. But science snatched victory from the jaws of defeat by moving the goal posts and redefining what is meant by a complete knowledge of the universe. ... In quantum theory, it turns out one doesn't need to know both the positions and the velocities [of the particles]. [22]

It is enough to know how the wave function evolves in time.

But the wave function is not directly observable. It cannot be measured by instruments. If deterministic behavior has no observer, does it lose its value? If, on the other hand, we define "behavior" in terms of measurable quantities, such as position and momentum, then the quantum model becomes nondeterministic, albeit in a subtle way. The wave function is commonly interpreted as giving the probabilities of the various possible inherently random measurement outcomes.

However, what could "probability" mean here? It turns out that it cannot be the usual notion of probability that you probably learned in an undergraduate class. Quantum probability is not an indicator of the relative frequency of various outcomes when performing repeated experiments. The so-called "no cloning theorem" in quantum mechanics says that such repeated experiments are impossible! The frequentist interpretation of probability has to be replaced by the Bayesian notion, which gives a different meaning to the word "probability" (see my book [38, Chapter 11] for a discussion of these two interpretations). In the Bayesian interpretation, a probability is a measure of what is unknown. Quantum probability is even stronger; it is a measure of what is *unknowable*.

It turns out that the mechanics of probability theory do not depend much on which interpretation you adopt. The math is the same. Interestingly, much of modern probability theory was developed by Laplace, who was convinced the world was completely deterministic. There is no contradiction here, however, because Laplace was a Bayesian. His probabilities measure uncertainty, or lack of knowledge, not intrinsic randomness.

Even if we take "behavior" to be the evolution of the wave function in time, despite our inability to measure it, subtleties remain. The Schrödinger equation gives the evolution of the wave function in classical Newtonian time, not relativistic time. Hence, even if we could measure the wave function, it would become possible for two distinct observers to see two different wave function evolutions for the same physical system! If the system includes metastable components, these distinct evolutions could be quite different indeed.

The role of an observer has always been important in quantum physics, albeit not in this relativistic sense. In what is now called the Copenhagen interpretation, originally proposed in the years 1925 to 1927 by Niels Bohr and Werner Heisenberg, the state of a system continues to be defined by probabilities until an external observer observes the state, and only at that point do the probabilities influence the outcome. Prior to being observed, all possible outcomes represented by the probabilities continue to remain possible. This requires an "observer" who is somehow separate from the system and measures the position of the particle. A "collapse" of the wave function occurs at the instant that a "measurement" is made, converting possibilities into certitude. This interpretation leaves unspecified what a "measurement" or "observation" is and typically puts the measurement apparatus outside the domain of quantum mechanics. This has led to sometimes bizarre interpretations, for example, that conscious minds play a central role, which presumably means that the physical world didn't exist or played by different rules before conscious minds formed.

The Copenhagen interpretation, however, is firmly rooted in an insistence on imposing a notion of "state" on the system. The outcome of a measurement is taken as a definitive answer, a fact about the system. A more fully "quantum" interpretation would continue to model the system, even after measurement, using a wave function.

In the 1950s, the physicist Hugh Everett III dispensed with the distinct observer, instead bundling observer and observed under a single wave function that evolves deterministically under the Schrödinger equation. The measurement apparatus and measured system entangle and evolve together in a single wave function. This view is a straightforward, simple, and direct interpretation of quantum mechanics until one insists on the same sort of certitude that one gets from the collapse of the wave function posited by the Copenhagen interpretation. With this insistence, the theory gets rather extravagant.

Consider a particle, say, an electron, moving through space. In classical physics, at each instant t, it has a definite position and momentum. In quantum physics, it has a wave function. If we measure its position, say, by putting a phosphorescent screen in its way, then the position of the electron will be revealed by a flash on the screen. Under the Copenhagen interpretation, at the time the electron hits the screen, its position is drawn randomly from a probability density function that puts weights equal to their likelihood on all regions of possible positions. Under Everett's interpretation, the photon, the screen, and the human observer become entangled and a single wave function covering all of them and continues to evolve deterministically.

But, under Everett's interpretation, where is the electron? Physicists seem to continue to insist that it must be somewhere, that it has "state," which leads to the most bizarre part of Everett's thesis. In his thesis, the electron is everywhere that its wave function permits it to be, but now in an uncountably infinite number of split-off universes. In each such universe, the electron is at one of the infinitely many possible positions. Each of these universes is somehow weighted by the probability dictated by the wave function.[11] The insistence on state leads to uncountably many universes being spawned anytime there is an interaction between components of a system.

[11]This is a rather mysterious part of the Many Worlds interpretation because the wave function defines a probability *density* for position, not a probability, so each of the uncountably many universes would have to have weight equal to zero. This stretches the notion of "existence" to the breaking point.

Because of the proliferation of universes, Everett's thesis is often called the "Many Worlds" interpretation of quantum mechanics. It means that every time there is an interaction between an observer and a subject, the universe splits. Depending on what is being measured, it could split in two or into an infinite number of possibilities. Since interactions are occurring all the time, a wildly extravagant proliferation of universes occurs at every instant in time. The model is deterministic, but the split universes cannot interact with one another, and hence, from the perspective of any entity in any one of those universes, the outcome of the experiment appears to be random.

Many physicists consider this consequence of Everett's thesis to be a *reductio ad absurdum* proof of the invalidity of the thesis. But others, including many highly respected physicists, accept the thesis as the best available explanation of quantum mechanics (see Becker [3] for a very readable overview of the alternative explanations).

There is, however, a simpler interpretation that is consistent with Everett's core idea that there is a single wave function that continues to evolve deterministically. The simpler interpretation rejects the notion that there is a ground truth about the state of a system at an instant in time. The electron is never at a definitive location. Despite seeing the flash of light, even a human observer is mistaken to interpret this as definitive evidence of a true location. No experimental apparatus is perfect, and it takes a great deal of effort to build an experimental apparatus that delivers even reasonable confidence, much less certainty. Moreover, no human perceptual system, which sees the flash, is perfect, and no brain has perfect memory. It is simply wrong to conclude that the flash is an indicator of a definitive truth. It is no more than compelling evidence of a high likelihood. This simpler interpretation has no need for a proliferation of universes. In exchange, it sacrifices certainty. Specifically, it sacrifices the notion of "state."

Einstein famously resisted aspects of quantum physics, a theory he helped to build, insisting that the theory was incomplete. He spent much of his later career searching for the "hidden variables," hitherto unknown aspects of the state of a physical system that would reconcile quantum theory with the principles of locality implied by relativity (that no event's consequences can propagate at faster than the speed of light). The conflict posed by quantum theory is between locality and a principle that some physicists call "reality." Here, "reality" is the principle that objects have real properties that determine the outcome of measurements, i.e., that they have what I have been calling a "state." Locality is the principle that reality at one location in space is not influenced instantaneously by measurements performed at a distant location (or more specifically, that influences propagate no faster than the speed of light).

In 1964, John Stewart Bell proved that quantum theory is incompatible with the principles of reality and locality [4]. That is, if quantum theory is a faithful scientific model of the physical world, something that is well verified experimentally, then one of the two principles must be false. Most physicists, including Bell himself, seem to prefer to sacrifice locality over reality.

However, this makes quantum theory incompatible with relativity, which requires locality. Relativity, however, does not require *reality* in this sense. In fact, as I argued in the previous section, relativity is inconsistent with reality in this sense. Relativity shows that a physical system, at least one that is spread out over space, cannot have "state" that, at an "instant" (a boundary between the past and the future), determines all of its properties. Any notion of state is dependent on the observer. The so-called "relational interpretation" of quantum mechanics, first attributed to the theoretical physicist Carlo Rovelli, takes this perspective. The state of a quantum system is a relation between the observer and the system. There is no notion of state that is independent of an observer (see Rovelli [64, 65] for readable explanations of this approach).

In summary, the world is not predictable under either classical or modern physics. Is it deterministic? That depends on what model we use, and all deterministic models have logical problems. A deterministic Newtonian model requires a presupposition of determinism. A deterministic

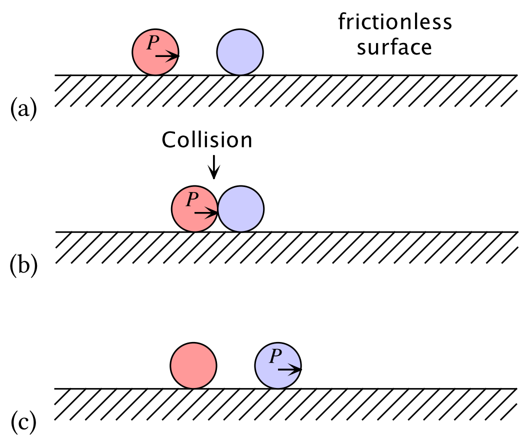

Fig. 4. Collision of ideal billiard balls on a frictionless surface.

relativistic model requires giving up the notion of state. A deterministic quantum model requires either an extravagant proliferation of universes or a similar forsaking of the notion of state. I will argue next that these logical problems are inevitable. Any set of deterministic models rich enough to say interesting things about the world is demonstrably incomplete, so these logical problems are unavoidable.

6 INCOMPLETENESS OF DETERMINISM

In 2016, I published a paper on the limits of modeling for cyber-physical systems in which I showed a sense in which determinism is incomplete [37] (see also my book [38, Chapter 10]). I review that result here. The short summary is that any set of deterministic models that is rich enough to encompass Newton's laws and also admits discrete behaviors does not contain its own limit points. Thus, any approach to modeling that relies on such a set of deterministic models has corner cases that exhibit nondeterminism. Nondeterminism, therefore, is inescapable in physical models.

6.1 Nondeterministic Collisions

Consider a model of the collisions of two billiard balls, as shown in Figure 4. Suppose that we model a collision as a discrete event, where the collision occurs in an instant, having no duration in time. Assume that the balls are ideally elastic, meaning that no kinetic energy is lost when they collide. In this case, Newton's laws require that both energy and momentum be conserved; the total momentum and energy must be the same after the collision as before.

Let v_1 and v_1' be the velocity of the first ball before and after the collision, respectively. Let v_2 and v_2' similarly represent the velocity of the second ball before and after the collision. Conservation of momentum requires

$$m_1 v_1' + m_2 v_2' = m_1 v_1 + m_2 v_2, \tag{3}$$

where m_1 and m_2 are the masses of the two balls, respectively. Conservation of kinetic energy requires

$$\frac{m_1(v_1')^2}{2} + \frac{m_2(v_2')^2}{2} = \frac{m_1(v_1)^2}{2} + \frac{m_2(v_2)^2}{2}.$$

Assuming we know the starting speeds v_1 and v_2 and the masses, then we have two equations and two unknowns, v_1' and v_2'. This is a quadratic problem with two solutions.

Solution 1: Ignore the collision:

$$v_1' = v_1,\ v_2' = v_2.$$

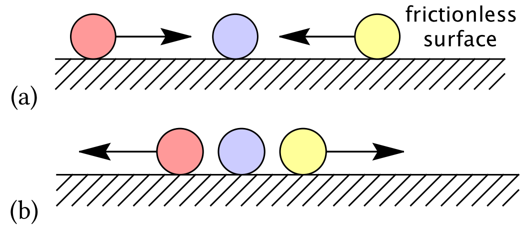

Fig. 5. Collision of three billiard balls on a frictionless surface.

Solution 2:

$$v_1' = \frac{v_1(m_1 - m_2) + 2m_2v_2}{m_1 + m_2}$$
$$v_2' = \frac{v_2(m_2 - m_1) + 2m_1v_1}{m_1 + m_2}.$$

Note that solution 1 looks like tunneling, where the balls pass through one another without affecting each other, as a neutrino might. In solution 2, if $m_1 = m_2$, then the two masses simply exchange velocities, as suggested in Figure 4. If we rule out solution 1, then the model is deterministic.

Now consider a more elaborate scenario, shown in Figure 5. Two (ideal) billiard balls approach a third stationary ball from opposite sides on a frictionless surface and collide with the stationary ball simultaneously. How should they react? If we consider only conservation of momentum and energy, we still have two equations, but now there are *three* unknowns.

At the time of the collision, there are *two* collisions. We could attempt to treat these as a pair of two-ball collisions, each of which has two possible outcomes. If we reject the tunneling solutions, then it would seem that only one outcome remains for each of the two collisions. But it is not obvious how to combine the two non-tunneling outcomes.

A first (naive) solution, using what is known as **Newton's hypothesis**, just superimposes the resulting momentums of the two non-tunneling outcomes. If the balls all have the same mass, then the left ball will transfer its momentum to the middle ball, the right ball will also transfer its momentum to the middle ball, and the equal and opposite momentums will cancel. All balls stop. Momentum is conserved, but not energy. This solution is shown at the top of Figure 6. Since there is no mechanism for energy dissipation in this model, this resolution is not satisfactory.

An alternative solution, using what is known in the literature as **Poisson's hypothesis**, introduces a form of superdense time [36]. At the time of the collision, the kinetic energy of the balls is instantly converted to potential energy by compressing the middle ball. Then, without time elapsing, in a second microstep, the potential energy is reconverted to kinetic energy by the middle ball expanding. But how should this ball apportion the kinetic energy to the two outer balls? An intuitively appealing solution is shown at the bottom of Figure 6, which assumes the masses are equal and the two balls bounce off the center ball and end up with equal and opposite velocities. In general, however, then there are many solutions that conserve both momentum and energy! We seem to have no basis for picking one solution over another, so the model appears to become intrinsically nondeterministic.

This thought experiment asked us to consider that the two collisions occur *simultaneously*. But, as we saw in the previous section, we have to choose an observer before simultaneity has any meaning. One observer may see the balls colliding simultaneously, another may see the left collision occurring first, and a third may see the right collision occurring first. It seems that any

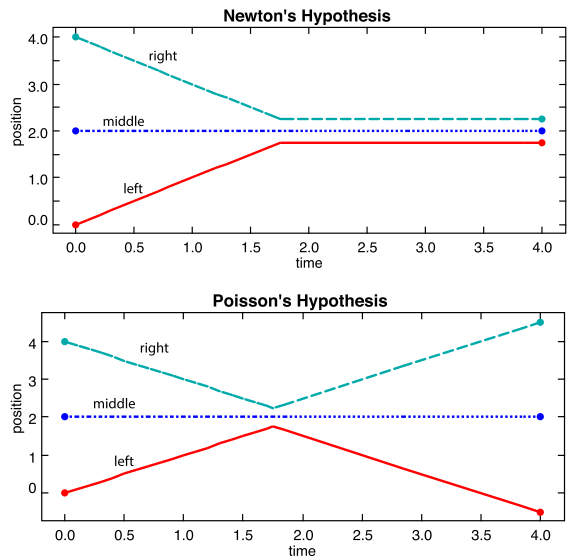

Fig. 6. Newton's hypothesis vs. Poisson's hypothesis.

solution we come up with needs to have some sort of consistency across the experiences of these three observers.

Let's consider what happens when we treat the collisions as occurring in some order, but without any time elapsing between the collisions (this again relies on a superdense model of time and can be seen as a limiting case where an infinitesimal amount of time elapses between the collisions). As shown in Figure 7, when the collisions occur, we arbitrarily pick either the left collision or the right one, temporarily ignoring the other one. Rejecting the tunneling solution, we get a deterministic exchange of momentum. Without time elapsing, we find ourself in state (b) in the figure, at which point we must handle the second collision. Again we get a deterministic solution, leaving us in state (c). Again, without time elapsing, we handle a third collision, which leaves us in state (d). After time elapses, we find ourselves in state (e).

I first studied this problem while developing the hybrid system simulator in Ptolemy II [7]. I experimented with different masses and with handling the collisions in different orders. I kept seeing plots like those shown in Figure 8. I was convinced that these plots should have been the same, that the order in which the software handled the collisions should not matter. Time did not elapse between collisions, so according to Newton's laws, the state of the system should not change. I spent weeks looking for the bug in the software before I finally realized that there was no bug in the software. More than one final state conserves both momentum and energy.

Inevitably, when I present this example, someone asks me how the "real world" behaves in this situation. This is a difficult question. One of the consequences of quantum mechanics is the Heisenberg uncertainty principle, which states that we cannot simultaneously know the position and momentum of an object to arbitrary precision. But discrete modeling of these collisions depends on modeling position and momentum precisely.

To many readers, it may seem odd to be invoking quantum mechanics on macro-scale physics problems, where Newtonian mechanics usually works just fine. But when the *order* of events affects the outcome, we find ourselves inevitably at quantum scales. The order of events can change with arbitrarily small differences in time or space.

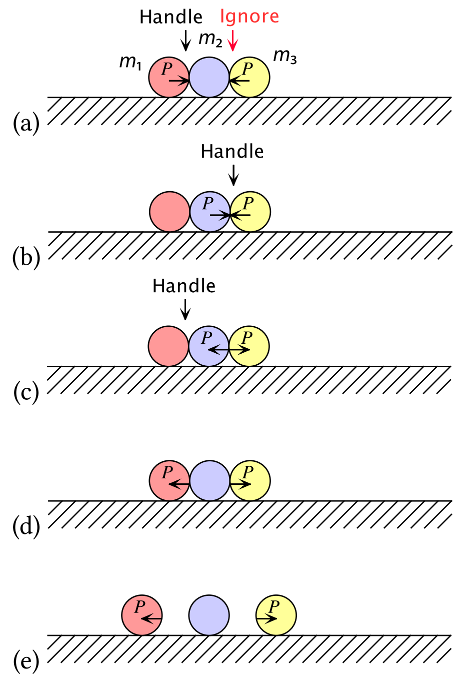

Fig. 7. One of two orderings for handling collisions.

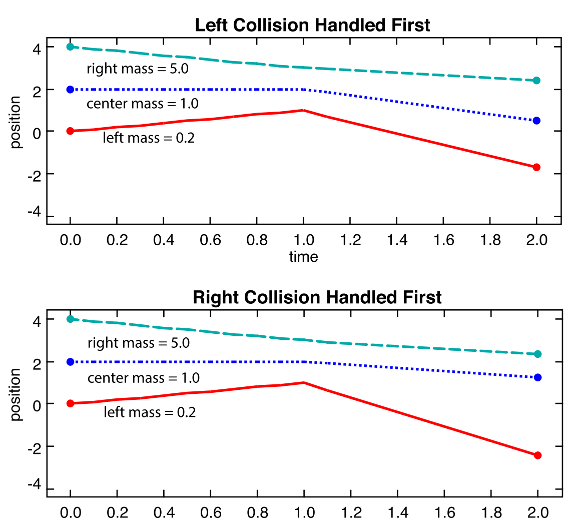

Fig. 8. If the masses are different, the behavior depends on which collision is handled first.

Both relativity and quantum physics expose difficulties with this mix of time and space continuums with discrete events. Perhaps it is the mix that is problematic and we should instead reject either continuums or discreteness. Rejecting continuums amounts to accepting a hypothesis sometimes called "digital physics," a position that I challenge in my book [38, Chapter 8].

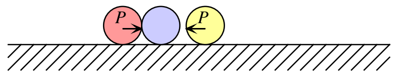

Fig. 9. Non-simultaneous collisions.

Rejecting discreteness is a bit harder to debunk, but it has the same flavor as the assumptions that Malament and Fletcher point out are required to rule out Norton's dome as a valid Newtonian system [20, 50]. Discrete collisions are singularities, not unlike the singularity that makes Norton's dome result in forces that are not continuously differentiable. But ruling them out simply to preserve determinism seems rather arbitrary. More disturbingly, I have developed an example (a flyback diode circuit) [37] that shows that rejecting discreteness has the consequence of also having to reject causality.

Nevertheless, for the billiard balls example, rejecting discreteness is relatively easy. Using only classical mechanics, we can model the balls as stiff springs, yielding a model where there are no discontinuous changes in momentum. I have shown that this makes the model deterministic, but chaotic [37]. Hence, arbitrarily small perturbations in initial conditions lead to arbitrarily large changes in final state. Combine this observation with quantum uncertainty, and the result is indistinguishable from nondeterminism in any practical sense.

6.2 Incompleteness

The three-billiard-ball example has the odd property that if the collisions are not simultaneous, then no matter how small the time difference between collisions is, the resulting behavior is deterministic. As the time between collisions approaches zero, as long as it remains nonzero, we have a deterministic model. But in the limit, the model becomes nondeterministic. This suggests that the set of deterministic models is incomplete, in that it does not contain its own limit points.

I review here a construction from my previous paper [37] of such an incomplete set of deterministic models. Consider the set M of models describing one-dimensional motion of $N = 3$ ideal elastic balls subject to Newton's second law, where collisions are handled with impulsive forces, and all behaviors that conserve momentum and energy are allowed except for tunneling. Every model in M is closed, in that there are no external forces, so all behaviors are a consequence of the initial conditions. To keep things simple, the set M includes three balls with specific masses $m_1 = 0.2$, $m_2 = 1.0$, and $m_3 = 5$ (these are the masses that generate the behaviors shown in Figure 8). The units do not matter, as long as they are consistent. Assume initial positions $x_i(0) \in \mathbb{R}$, and initial velocities $v_i(0) \in \mathbb{R}$ given as follows:

$$\begin{aligned} x_1(0) &= -1 \\ v_1(0) &= 1 \\ x_2(0) &= 0 \\ v_2(0) &= 0 \\ x_3(0) &= 1 + \Delta \\ v_3(0) &= -1, \end{aligned}$$

where $\Delta \in \mathbb{R}$ is a real number. Δ is the only parameter that distinguishes models in M, but this one variable is sufficient to give us an uncountably infinite set of models. When $\Delta \neq 0$, the collisions will not be simultaneous, as illustrated in Figure 9.

Following Lee and Sangiovanni-Vincentelli [42], we formally define a model as a set of behaviors. It is sufficient to consider the behavior of each model over the time interval $[0, 2]$ only. Let B be the set of all functions of the form $b : [0, 2] \to \mathbb{R}^3$. For a particular model $A \in M$ with parameter Δ_A, we say that $x \in B$ is a behavior of A if for all $t \in [0, 2]$, $x_i(t)$ is the position in our one-dimensional space of ball i at time t, for $i \in \{1, 2, 3\}$. These x_i are functions of time that satisfy the equations of motion and conservation laws. In other words, a behavior of a model $A \in M$ is a function x giving the positions of the three balls as a function of time.

If a particular model $A \in M$ has exactly one behavior, then A is deterministic. As shown above, M contains only one nondeterministic model, let's call it $N \in M$, the one where $\Delta_N = 0$. We can now construct a sequence of deterministic models in D that should converge to N but does not. To do that, we need some notion of convergence.

Consider the subset $D \subset M$ of deterministic models. We can define a metric on D that allows us to talk about models being "close." Consider two models $A, A' \in D$. Because they are in D, these models are deterministic. Each admits one behavior, $x, x' \in B$, respectively. Because each has exactly one behavior, we can define a distance function d as follows:

$$d(A, A') = \frac{1}{2} \int_0^2 ||x(t) - x'(t)||dt, \tag{4}$$

where $||x||$ is the L1 norm of a real vector x. This distance function measures the difference between two ball trajectories. It is easy to show that d is a metric, and hence (D, d) is a metric space.

Consider a sequence of models $A_i \in D$, $i \in \{1, 2, \ldots\}$ where

$$\Delta_{A_i} = 1/i^2.$$

As i gets larger, the time between collisions gets smaller, so in some sense, these models approximate the nondeterministic case N where $\Delta_N = 0$ ever more closely as i gets larger. It is easy to show that the sequence A_i is Cauchy, which means that for any $\epsilon > 0$, we can find a positive integer N such that for all positive integers $i, j > N$,

$$d(A_i, A_j) < \epsilon.$$

This means that as i and j get large, the trajectories of the balls in A_i get ever closer to the trajectories of the balls in A_j. With sufficiently large i and j, A_i becomes nearly indistinguishable from A_j. Nevertheless, this sequence has no limit in D.

The one nondeterministic model $N \in M$, where the collisions are exactly simultaneous, $\Delta_N = 0$, admits behaviors that are very distant from the behaviors of the models in the sequence A_i using the same metric. Some of the behaviors of N will be close (in this metric space sense) to behaviors of a very different Cauchy sequence, $C_i \in D$, $i \in \{1, 2, \ldots\}$, where

$$\Delta_{C_i} = -1/i^2.$$

Here, the time between collisions is approaching zero from the other side.

A metric space that has Cauchy sequences that have no limit in the space is said to be "incomplete." The metric space (D, d) of deterministic models is incomplete. It does not contain all its limit points.

Every model in the sequence A_i is deterministic, and the models in the sequence get arbitrarily close to one another. Moreover, any set of deterministic models rich enough to encompass Newton's laws that allows discrete collisions must be rich enough to include this sequence of models, and therefore will be incomplete.

This has profound consequences that I explore in my book [38]. Specifically, many people assume without proof that if some modeling technique can be shown to be able to arbitrarily closely approximate any model of interest, then this technique is "good enough" for all practical purposes. This example shows that this assumption is untrue unless the modeling technique defines a complete set of models. Our Cauchy sequence A_i arguably represents an arbitrarily close approximation to N, and yet it fails to capture a hugely important property of N, that it is nondeterministic.

7 CONCLUSION

I hope I have convinced you that determinism is a deep subject. In reality, I have only scratched the surface here. In my book, *The Coevolution*, I explore an even harder aspect, the connection between determinism and free will [39, Chapter 12]. (Hint: observers become important again.) The focus there is to try to get a handle on the question of whether humans will ever build machines that have, in any sense, free will. More practically, the question is whether we humans will ever build machines that can and should be held accountable for their actions. That question is beyond the scope of this article.

This article focuses instead on the practical question of the usefulness and completeness of deterministic models. I have argued that they are extremely useful, and that even though nondeterministic models also have their uses, those uses are disjoint. The choice of whether to use deterministic or nondeterministic models should be front and center in any engineering design effort, and yet it rarely is. Whether the models are deterministic or not typically depends on the choice of modeling and design frameworks, and many of these are nondeterministic by accident, not by intent.

I have shown that whether a model is deterministic or not depends on how one defines the inputs and behavior of the model. To define behavior, one has to define an observer. I compared and contrasted two classes of ways to do it, one based on the notion of "state" and another that more flexibly defines the observables. A state-based model further requires a restrictive model of time, one that is known to be an inaccurate model of physical time. Specifically, it requires an unambiguous simultaneity.

I examine determinism in models of the physical world. In what may surprise many readers, I show that Newtonian physics admits nondeterminism (unless one presupposes determinism) and that quantum physics may be interpreted as a deterministic model. Moreover, I show that both relativity and quantum mechanics undermine the notion of "state" and therefore require the more flexible ways of defining observables. Finally, I show that sufficiently rich sets of deterministic models are incomplete, Hence, no matter how much we value them, they will not solve all our problems.

In engineering practice, whether to use nondeterministic models becomes a central question. For scientific models, where it is incumbent on the model to match the thing being modeled, nondeterminism can be useful if it reflects inherent randomness or uncertainty about the thing being modeled. In this case, nondeterminism enables exploration of the range of possible behaviors. However, nondeterminism says nothing about the likelihood of any behavior, only about its possibility. Hence, a probabilistic model may be more useful than a nondeterministic one. A probabilistic model may usefully model a deterministic system (by interpreting probabilities as a measure of uncertainty) and may even, in some circumstances, be interpreted as a deterministic model (as we can do with quantum physics).

For engineering models, where it is incumbent on the thing being modeled to match the model, nondeterminism can be a useful abstraction mechanism, a way to get simpler models for analysis. It can also be useful for deferring design decisions. However, it comes at a steep price. The model becomes less useful for testing, for evaluating the degree to which the thing being modeled matches the model. Nondeterminism that arises from sloppiness in the modeling framework or language, however, is rarely useful and should not be excused by the fact that the physical world is unpredictable in practice.

My own focus as an engineer is on cyber-physical systems, which combine the neat world of computation with the messy and unpredictable physical world. Deterministic models play an important role on both sides of this divide. For example, on the physical side, deterministic differential equation models can be useful descriptions of how a robot arm *should* behave, particularly if coupled with probabilistic models of how it may actually behave. On the cyber side, for distributed software, clear specifications of how the components *should* coordinate are useful, particularly if coupled with probabilistic models of network behavior that may compromise these specifications. In both cases, we are talking about a combination of engineering and scientific models.

Today, engineers settle for much less than this. For example, in robotics, **the Robot Operating System (ROS)** [62] is popular for coordinating disparate software components, but it uses a publish-and-subscribe communication fabric that is unnecessarily nondeterministic [48]. In the context of the **Internet of Things (IoT)**, MQTT [25] is a popular coordination mechanism that is similarly based on publish-and-subscribe. The latest and greatest in automotive software, Adaptive AUTOSAR [2], is also unnecessarily nondeterministic [52]. In industrial automation, the standards governing **Programmable Logic Controller (PLC)** software also admit nondeterminsm [69]. The actor-based communication fabrics of Erlang [1], Akka [63], and Ray [56] are similarly nondeterministic [47]. In all of these frameworks, it is *possible* to construct deterministic models, but it requires considerable expertise in concurrent software, expertise well outside the comfort zone of most application engineers. So the application engineers accept the nondeterminism, tweak priorities, insert ad hoc delays, test their implementations extensively, and hope for the best. We can do better.

ACKNOWLEDGMENTS

The author thanks Gérard Berry, Stephen Edwards, Shaokai Lin, Marten Lohstroh, John Norton, Sanjit Seshia, Marjan Sirjani, Reinhard von Hanxleden, Jean Walrand, and the anonymous reviewers for useful suggestions based on an earlier draft.

REFERENCES

[1] Joe Armstrong, Robert Virding, Claes Wikström, and Mike Williams. 1996. *Concurrent Programming in Erlang* (2nd ed.). Prentice Hall.

[2] AUTOSAR. 2019. Explanation of adaptive platform design. *AUTOSAR AP Release 19-11*.

[3] Adam Becker. 2018. *What Is Real? The Unfinished Quest for the Meaning of Quantum Physics*. Basic Books.

[4] John Stewart Bell. 1964. On the Einstein Podolsky Rosen paradox. *Physica* 1, 3 (1964), 195–200.

[5] Albert Benveniste and Gérard Berry. 1991. The synchronous approach to reactive and real-time systems. *Proceedings of IEEE* 79, 9 (1991), 1270–1282.

[6] George E. P. Box and Norman R. Draper. 1987. *Empirical Model-Building and Response Surfaces*. Wiley.

[7] Janette Cardoso, Edward A. Lee, Jie Liu, and Haiyang Zheng. 2014. Continuous-time models. In *System Design, Modeling, and Simulation using Ptolemy II*, Claudius Ptolemaeus (Ed.). Ptolemy.org, Berkeley, CA. http://ptolemy.org/books/Systems.

[8] Alonzo Church. 1932. A set of postulates for the foundation of logic. *Annals of Mathematics* 32, 2 (April 1932), 346–366. https://www.jstor.org/stable/1968337.

[9] Alonzo Church and J. B. Rosser. 1936. Some properties of conversion. *Transactions of the American Mathematical Society* 39, 3 (May 1936), 472–482. DOI: https://doi.org/10.2307/1989762

[10] Bridget Copley and Fabienne Martin. 2014. *Causation in Grammatical Structures*. Oxford University Press, Oxford, England.

[11] James C. Corbett, Jeffrey Dean, Michael Epstein, Andrew Fikes, Christopher Frost, J. J. Furman, Sanjay Ghemawat, Andrey Gubarev, Christopher Heiser, Peter Hochschild, Wilson Hsieh, Sebastian Kanthak, Eugene Kogan, Hongyi Li, Alexander Lloyd, Sergey Melnik, David Mwaura, David Nagle, Sean Quinlan, Rajesh Rao, Lindsay Rolig, Yasushi Saito, Michal Szymaniak, Christopher Taylor, Ruth Wang, and Dale Woodford. 2012. Spanner: Google's globally-distributed database. In *USENIX Symposium on Operating Systems Design and Implementation (OSDI'12)*. DOI: https://doi.org/10.1145/2491245

[12] Patrick Cousot and Radhia Cousot. 1977. Abstract interpretation: A unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *Symposium on Principles of Programming Languages (POPL'77)*. ACM Press, 238–252.

[13] Fabio Cremona, Marten Lohstroh, David Broman, Edward A. Lee, Michael Masin, and Stavros Tripakis. 2017. Hybrid co-simulation: It's about time. *Software and Systems Modeling* 18 (November 2017), 1655–1679. DOI: https://doi.org/10.1007/s10270-017-0633-6

[14] Abhishek Dhar. 1993. Nonuniqueness in the solutions of Newton's equation of motion. *American Journal of Physics* 61, 1 (1993), 58–61. DOI: https://doi.org/10.1119/1.17411

[15] Susanne Ditlevsen and Adeline Samson. 2013. Introduction to stochastic models in biology. In *Stochastic Biomathematical Models: With Applications to Neuronal Modeling*, Mostafa Bachar, Jerry Batzel, and Susanne Ditlevsen (Eds.). Springer, 3–35. DOI: https://doi.org/10.1007/978-3-642-32157-3_1

[16] John Earman. 1986. *A Primer on Determinism*. The University of Ontario Series in Philosophy of Science, Vol. 32. D. Reidel Publishing Company.

[17] Stephen Edwards. 2018. *On Determinism*. Vol. LNCS 10760. Springer, Cham, Switzerland, 240–253. DOI: https://doi.org/10.1007/978-3-319-95246-8_14

[18] Stephen Edwards and John Hui. 2020. The sparse synchronous model. In *2020 Forum for Specification and Design Languages (FDL'20)*. IEEE, 1–8.

[19] Stephen A. Edwards and Edward A. Lee. 2003. The semantics and execution of a synchronous block-diagram language. *Science of Computer Programming* 48, 1 (2003), 21–42.

[20] Samuel Craig Fletcher. 2012. What counts as a Newtonian system? The view from Norton's dome. *European Journal for Philosophy of Science* 2 (October 2012), 275–297. DOI: https://doi.org/10.1007/s13194-011-0040-8

[21] Richard Gale. 1966. McTaggart's analysis of time. *American Philosophical Quarterly* 3, 2 (1966), 145–152. https://www.jstor.org/stable/20009201.

[22] Stephen Hawking. 2002. Gödel and the end of the universe. *Stephen Hawking Public Lectures*. http://www.hawking.org.uk/godel-and-the-end-of-physics.html.

[23] Carl Hoefer. 2016. Causal determinism. *The Stanford Encyclopedia of Philosophy*.Spring 2016 Edition. http://plato.stanford.edu/archives/spr2016/entries/determinism-causal/.

[24] John Hopcroft and Jeffrey Ullman. 1979. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley, Reading, MA.

[25] Urs Hunkeler, Hong Linh Truong, and Andy Stanford-Clark. 2008. MQTT-S—A publish/subscribe protocol for wireless sensor networks. In *3rd International Conference on Communication Systems Software and Middleware and Workshops (COMSWARE'08)*. IEEE, 791–798.

[26] Gilles Kahn. 1974. The semantics of a simple language for parallel programming. In *Proceedings of the IFIP Congress 74*. North-Holland Publishing Co., 471–475.

[27] Gilles Kahn and D. B. MacQueen. 1977. Coroutines and networks of parallel processes. In *Information Processing*, B. Gilchrist (Ed.). North-Holland Publishing Co., 993–998.

[28] Victor Khomenko, Mark Schaefer, and Walter Vogler. 2008. Output-determinacy and asynchronous circuit synthesis. *Fundamenta Informaticae* 88, 4 (2008), 541–579.

[29] David J. Kinniment. 2007. *Synchronization and Arbitration in Digital Systems*. John Wiley & Sons, Ltd., New York.

[30] Pierre-Simon Laplace. 1901. *A Philosophical Essay on Probabilities*. John Wiley and Sons, Hoboken, NJ. Translated from the 6th French edition by F. W. Truscott and F. L. Emory.

[31] Edward A. Lee. 1999. Modeling concurrent real-time processes using discrete events. *Annals of Software Engineering* 7 (1999), 25–45.

[32] Edward A. Lee. 2006. Concurrent semantics without the notions of state or state transitions. In *International Conference on Formal Modelling and Analysis of Timed Systems (FORMATS'06)*, E. Asarin and P. Bouyer (Eds.), Vol. LNCS 4202. Springer-Verlag. DOI: https://doi.org/10.1007/11867340_2

[33] Edward A. Lee. 2006. The problem with threads. *Computer* 39, 5 (2006), 33–42.

[34] Edward A. Lee. 2008. Cyber physical systems: Design challenges. In *International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing (ISORC'08)*. IEEE, 363–369. DOI: https://doi.org/10.1109/ISORC.2008.25

[35] Edward A. Lee. 2009. Computing needs time. *Communications of the ACM* 52, 5 (2009), 70–79. DOI: https://doi.org/10.1145/1506409.1506426

[36] Edward A. Lee. 2014. Constructive models of discrete and continuous physical phenomena. *IEEE Access* 2, 1 (2014), 1–25. DOI: https://doi.org/10.1109/ACCESS.2014.2345759

[37] Edward A. Lee. 2016. Fundamental limits of cyber-physical systems modeling. *ACM Transactions on Cyber-Physical Systems* 1, 1 (2016), 26. DOI: https://doi.org/10.1145/2912149

[38] Edward Ashford Lee. 2017. *Plato and the Nerd — The Creative Partnership of Humans and Technology*. MIT Press, Cambridge, MA.

[39] Edward Ashford Lee. 2020. *The Coevolution: The Entwined Futures of Humans and Machines*. MIT Press, Cambridge, MA.

[40] Edward A. Lee and Eleftherios Matsikoudis. 2009. The semantics of dataflow with firing. In *From Semantics to Computer Science: Essays in Memory of Gilles Kahn*, Gérard Huet, Gordon Plotkin, Jean-Jacques Lévy, and Yves Bertot (Eds.). Cambridge University Press.

[41] E. A. Lee and T. M. Parks. 1995. Dataflow process networks. *Proceedings of IEEE* 83, 5 (1995), 773–801. DOI: https://doi.org/10.1109/5.381846

[42] Edward A. Lee and Alberto Sangiovanni-Vincentelli. 1998. A framework for comparing models of computation. *IEEE Transactions on Computer-Aided Design of Circuits and Systems* 17, 12 (1998), 1217–1229.

[43] Edward A. Lee and Marjan Sirjani. 2018. What good are models? In *Formal Aspects of Component Software (FACS'18)*, Vol. LNCS 11222. Springer.

[44] Edward A. Lee and Haiyang Zheng. 2007. Leveraging synchronous language principles for heterogeneous modeling and design of embedded systems. In *EMSOFT*. ACM, 114–123.

[45] Edwin R. Lewis and Ronald J. MacGregor. 2006. On indeterminism, chaos, and small number particle systems in the brain. *Journal of Integrative Neuroscience* 5, 2 (2006), 223–247. https://people.eecs.berkeley.edu/~lewis/LewisMacGregor.pdf.

[46] Marten Lohstroh, Íñigo Íncer Romeo, Andrés Goens, Patricia Derler, Jeronimo Castrillon, Edward A. Lee, and Alberto Sangiovanni-Vincentelli. 2019. Reactors: A deterministic model for composable reactive systems. In *8th International Workshop on Model-Based Design of Cyber Physical Systems (CyPhy'19)*, Vol. LNCS 11971. Springer-Verlag. DOI: https://doi.org/10.1007/978-3-030-41131-2_4

[47] M. Lohstroh and E. A. Lee. 2019. Deterministic actors. In *2019 Forum for Specification and Design Languages (FDL'19)*. 1–8. DOI: https://doi.org/10.1109/FDL.2019.8876922

[48] Marten Lohstroh, Christian Menard, Alexander Schulz-Rosengarten, Matthew Weber, Jeronimo Castrillon, and Edward A. Lee. 2020. A language for deterministic coordination across multiple timelines. In *Forum for Specification and Design Languages (FDL'20)*. IEEE. DOI: https://doi.org/10.1109/FDL50818.2020.9232939

[49] Edward N. Lorenz. 1963. Deterministic nonperiodic flow. *Journal of the Atmospheric Sciences* 20 (1963), 130–141. DOI: https://doi.org/10.1175/1520-0469(1963)020⟨0130:DNF⟩2.0.CO;2

[50] David B. Malament. 2008. Norton's slippery slope. *Philosophy of Science* 75 (December 2008), 799–816. DOI: https://doi.org/10.1086/594525

[51] Leonard R. Marino. 1981. General theory of metastable operation. *IEEE Transactions on Computers* C-30, 2 (1981), 107–115.

[52] Christian Menard, Andrés Goens, Marten Lohstroh, and Jeronimo Castrillon. 2020. Achieving derterminism in adaptive AUTOSAR. In *Design, Automation and Test in Europe (DATE'20)*. Grenoble, France.

[53] Michael Mendler, Thomas R. Shiple, and Gérard Berry. 2012. Constructive Boolean circuits and the exactness of timed ternary simulation. *Formal Methods in System Design* 40, 3 (2012), 283–329. DOI: https://doi.org/10.1007/s10703-012-0144-6

[54] Robin Milner. 1980. *A Calculus of Communicating Systems*. Lecture Notes in Computer Science, Vol. 92. Springer.

[55] Robin Milner. 1989. *Communication and Concurrency*. Prentice Hall, Englewood Cliffs, NJ.

[56] Philipp Moritz, Robert Nishihara, Stephanie Wang, Alexey Tumanov, Richard Liaw, Eric Liang, William Paul, Michael I. Jordan, and Ion Stoica. 2017. Ray: A distributed framework for emerging AI applications. *CoRR* abs/1712.05889 (2017). arxiv:1712.05889.

[57] John D. Norton. 2007. Causation as folk science. In *Causation, Physics, and the Constitution of Reality*, Huw Price and Richard Corry (Eds.). Clarendon Press, Oxford, Book section 2, 11–44.

[58] David Park. 1980. Concurrency and automata on infinite sequences. In *Theoretical Computer Science*, Deussen P. (Ed.), Vol. LNCS 104. Springer, Berlin. DOI: https://doi.org/10.1007/BFb0017309

[59] David A. Patterson, Garth Gibson, and Randy Katz. 1988. A case for redundant arrays of inexpensive disks (RAID). In *International Conference on Management of Data (SIGMOD'88)*. DOI: https://doi.org/10.1145/50202.50214

[60] Judea Pearl and Dana Mackenzie. 2018. *The Book of Why: The New Science of Cause and Effect*. Basic Books, New York.

[61] Karl Popper. 1959. *The Logic of Scientific Discovery*. Hutchinson & Co., London and New York. Taylor & Francis edition, 2005.

[62] Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Ng. 2009. ROS: An open-source robot operating system. In *International Conference on Robotics and Automation (ICRA'09) Workshop on Open Source Software* 3.

[63] Raymond Roestenburg, Rob Bakker, and Rob Williams. 2016. *Akka In Action*. Manning Publications Co.

[64] Carlo Rovelli. 2017. *Reality Is Not What It Seems: The Journey to Quantum Gravity*. Riverhead Books, New York.

[65] Carlo Rovelli. 2018. *The Order of Time*. Riverhead Books, New York.

[66] Bertrand Russell. 1913. On the notion of cause. *Proceedings of the Aristotelian Society* 13 (1913), 1–26.

[67] Davide Sangiorgi. 2009. On the origins of bisimulation and coinduction. *ACM Transactions on Programming Languages and Systems* 31, 4 (2009), 15:1–15:41. DOI: https://doi.org/10.1145/1516507.1516510

[68] Alexander Schulz-Rosengarten, Reinhard von Hanxleden, Frédéric Mallet, Robert De Simone, and Julien Deantoni. 2018. Time in SCCharts. In *2018 Forum on Specification Design Languages (FDL'18)*. 5–16.

[69] Martin A. Sehr, Marten Lohstroh, Mathew Weber, Ines Ugalde, Martin Witte, Joerg Neidig, Stephan Hoeme, Mehrdad Niknami, and Edward A. Lee. 2021. Programmable logic controllers in the context of industry 4.0. *IEEE Transactions on Industrial Informatics* 17, 5 (May 2021), 3523–3533. DOI: https://doi.org/10.1109/TII.2020.3007764
[70] A. M. Turing. 1936. On computable numbers with an application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society* 42 (1936), 230–265.
[71] Reinhard von Hanxleden et al. 2014. SCCharts: Sequentially constructive Statecharts for safety-critical applications. In *ACM SIGPLAN Conf. on Programming Language Design and Implementation (PLDI'14)*. ACM, New York, NY, 372–383. DOI: https://doi.org/10.1145/2594291.2594310
[72] Reinhard von Hanxleden, Timothy Bourke, and Alain Girault. 2017. Real-time ticks for synchronous programming. In *2017 Forum on Specification and Design Languages (FDL'17)*. IEEE, 1–8.
[73] G. Winskel. 1993. *The Formal Semantics of Programming Languages*. MIT Press, Cambridge, MA.
[74] Remigiusz Wisniewski, Iwona Grobelna, and Andrei Karatkevich. 2020. Determinism in cyber-physical systems specified by interpreted petri nets. *Sensors* 20, 5565 (2020), 22. DOI: https://doi.org/10.3390/s20195565
[75] Stephen Wolfram. 2002. *A New Kind of Science*. Wolfram Media, Inc.
[76] David H. Wolpert. 2008. Physical limits of inference. *Physica* 237, 9 (2008), 1257–1281. DOI: https://doi.org/10.1016/j.physd.2008.03.040
[77] Yang Zhao, Edward A. Lee, and Jie Liu. 2007. A programming model for time-synchronized distributed real-time systems. In *Real-Time and Embedded Technology and Applications Symposium (RTAS'07)*. IEEE, 259–268.
[78] Henrik Zinkernagel. 2010. *Causal Fundamentalism in Physics*. Springer, Dordrecht.

Received January 2021; revised March 2021; accepted March 2021