



Resurrecting Laplace's Demon: The Case for Deterministic Models

Edward A. Lee

*Robert S. Pepper Distinguished Professor
UC Berkeley*

Keynote Talk: MODELS

October 4, 2016

St. Malo, France



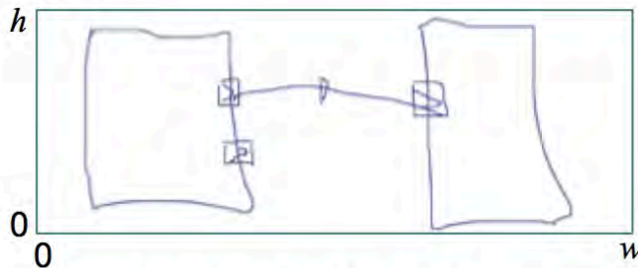


MODELS 2010

Oslo, Norway

Keynote talk: *Disciplined Heterogeneous Modeling*

The Truly Unified Modeling Language *TUML*



A model in TUML is a function of the form

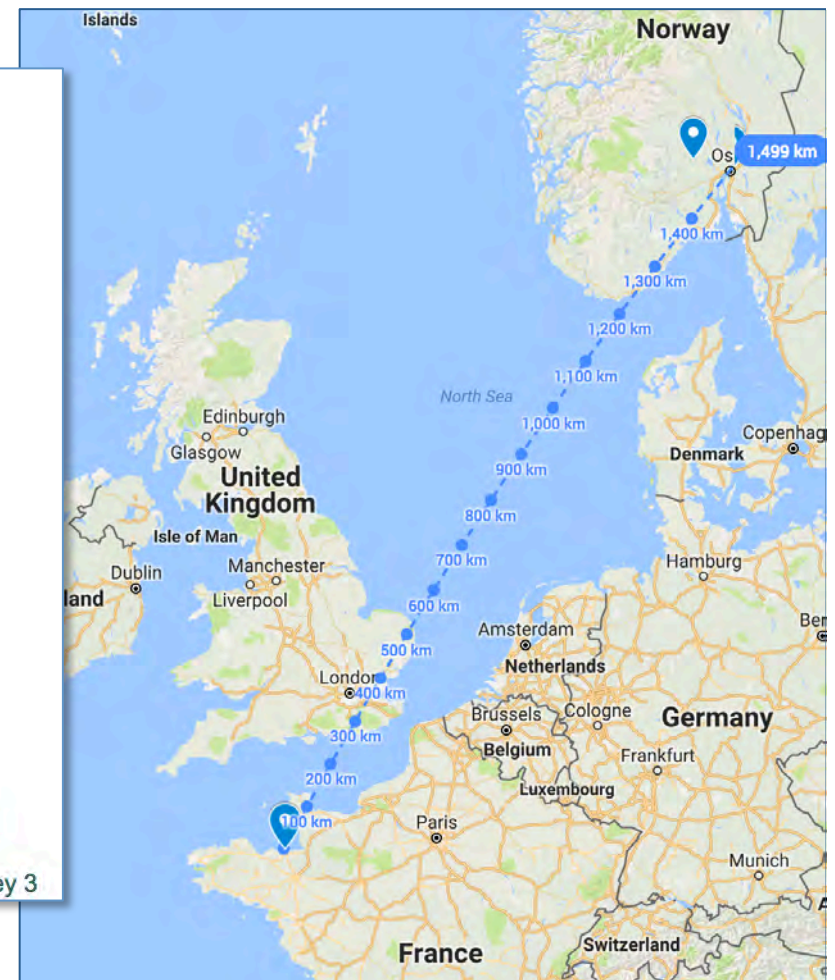
$$f: [0, w] \times [0, h] \rightarrow \{0, 1\}, \quad w, h \in \mathbb{N}$$

(notice how nicely formal the language is!)

Tools already exist.

With the mere addition of a *TUML profile*, every existing UML notation is a special case!

Lee, Berkeley 3



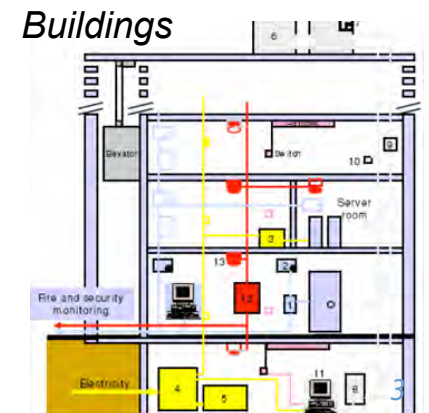
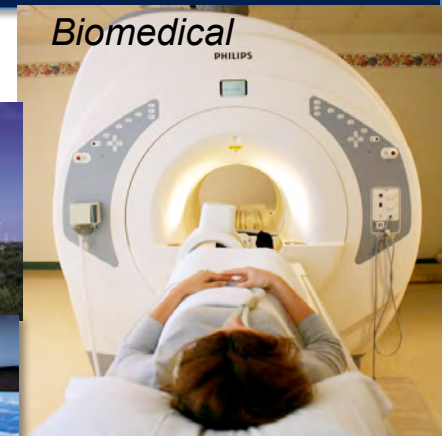
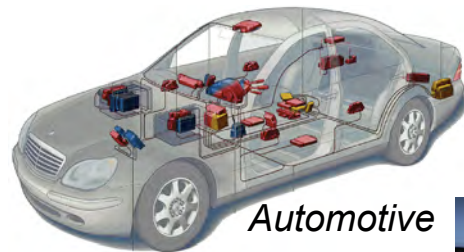


Context: Cyber-Physical Systems

A particularly challenging case for determinism

Not just information technology:

- Cyber + *Physical*
- Computation + *Dynamics*
- Security + *Safety*



Properties:

- Highly dynamic
- Safety critical
- Uncertain environment
- Physically distributed
- Sporadic connectivity
- Resource constrained

Does it make sense to talk about deterministic models for such systems?

Lee, Berkeley



Models vs. Reality

$$x(t) = x(0) + \int_0^t v(\tau) d\tau$$
$$v(t) = v(0) + \frac{1}{m} \int_0^t F(\tau) d\tau.$$

The model

In this example, the *modeling framework* is calculus and Newton's laws.



The target
(the thing
being
modeled).

Fidelity is how well the model and its target match



Engineers often confuse the model with its target

You will never strike oil by drilling through the map!

But this does not in any way diminish the value of a map!



Solomon Wolf Golomb

Lee, Berkeley





Determinacy

Some of the most valuable models are *deterministic*.

A model is *deterministic* if, given the initial *state* and the *inputs*, the model defines exactly one *behavior*.

Deterministic models have proven extremely valuable in the past.



Laplace's Demon

“We may regard the present state of the universe as the effect of its past and the cause of its future. An intellect which at a certain moment would know all forces that set nature in motion, and all positions of all items of which nature is composed, if this intellect were also vast enough to submit these data to analysis, it would embrace in a single formula the movements of the greatest bodies of the universe and those of the tiniest atom; for such an intellect nothing would be uncertain and the future just like the past would be present before its eyes.”

— *Pierre Simon Laplace*

*Pierre-Simon Laplace (1749–1827).
Portrait by Joan-Baptiste Paulin Guérin, 1838*

Lee, Berkeley





Did quantum mechanics dash this hope?

“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.”

(Stephen Hawking, 2002)





Nevertheless, Laplace's Demon cannot exist.

In 2008, David Wolpert, then at NASA, now at the Santa Fe Research Institute, used Cantor's diagonalization technique to prove that Laplace's demon cannot exist. His 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.



David Wolpert



The Koptez Principle



Hermann Kopetz
Professor (Emeritus)
TU Vienna

Many properties that we assert about systems (determinism, timeliness, reliability) are in fact not properties of the system, but rather properties of a *model* of the system.

If we accept this, then it makes no sense to talk about whether the physical world is deterministic. It only makes sense to talk about whether *models* of the physical world are deterministic.



The question switches from whether a model is *True* to whether it is *Useful*

“Essentially, all models are wrong,
but some are useful.”

Box, G. E. P. and N. R. Draper, 1987: *Empirical Model-Building and Response Surfaces*. Wiley Series in Probability and Statistics, Wiley.



Physicists continue to debate whether the world is deterministic

$$x(t) = x(0) + \int_0^t v(\tau) d\tau$$
$$v(t) = v(0) + \frac{1}{m} \int_0^t F(\tau) d\tau.$$

Deterministic model

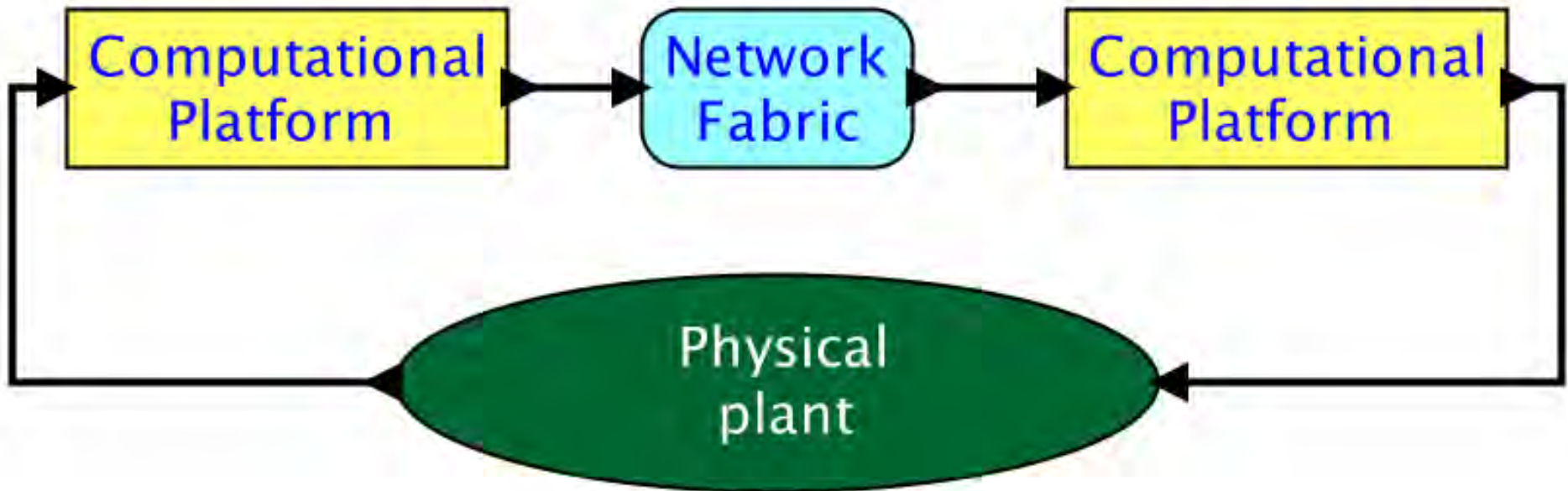


Deterministic system?

Determinism is a property of models, not a property of the systems they model.



Schematic of a simple Cyber-Physical System



What kinds of models should we use?

Let's look at the most successful kinds of models from the cyber and the physical worlds.



Software is a Model

Physical System



Model

```
/** Reset the output receivers, which are the inside receivers of
 * the output ports of the container.
 * @exception IllegalArgumentException If getting the receivers fails.
 */
private void _resetOutputReceivers() throws IllegalArgumentException {
    List<IOPort> outputs = ((Actor) getContainer()).outputPortList();
    for (IOPort output : outputs) {
        if (_debugging) {
            _debug("Resetting inside receivers of output port: "
                + output.getName());
        }
        Receiver[] receivers = output.getInsideReceivers();
        if (receivers != null) {
            for (int i = 0; i < receivers.length; i++) {
                if (receivers[i] != null) {
                    for (int j = 0; j < receivers[i].length; j++) {
                        if (receivers[i][j] instanceof FSMReceiver) {
                            receivers[i][j].reset();
                        }
                    }
                }
            }
        }
    }
}
```

*Single-threaded imperative programs
are deterministic models*



Consider single-threaded imperative programs

```
1 void foo(int32_t x) {  
2     if (x > 1000) {  
3         x = 1000;  
4     }  
5     if (x > 0) {  
6         x = x + 1000;  
7         if (x < 0) {  
8             panic();  
9         }  
10    }  
11 }
```

This program defines exactly one behavior, given the input x.

Note that the modeling framework (the C language, in this case) defines “behavior” and “input.”



The target of the model is electrons sloshing around in silicon. It takes time, consumes energy, and fails if dropped in the ocean, none of which are properties of the model.



Software relies on another deterministic model that abstracts the hardware

Physical System

Model



Image: Wikimedia Commons

Integer Register-Register Operations

RISC-V defines several arithmetic R-type operations. All operations read the *rs1* and *rs2* registers as source operands and write the result into register *rd*. The *funct* field selects the type of operation.

31	27 26	22 21	17 16	7 6	0
rd	rs1	rs2	funct10	opcode	
5	5	5	10	7	
dest	src1	src2	ADD/SUB/SLT/SLTU	OP	
dest	src1	src2	AND/OR/XOR	OP	
dest	src1	src2	SLL/SRL/SRA	OP	
dest	src1	src2	ADDW/SUBW	OP-32	
dest	src1	src2	SLLW/SRLW/SRAW	OP-32	

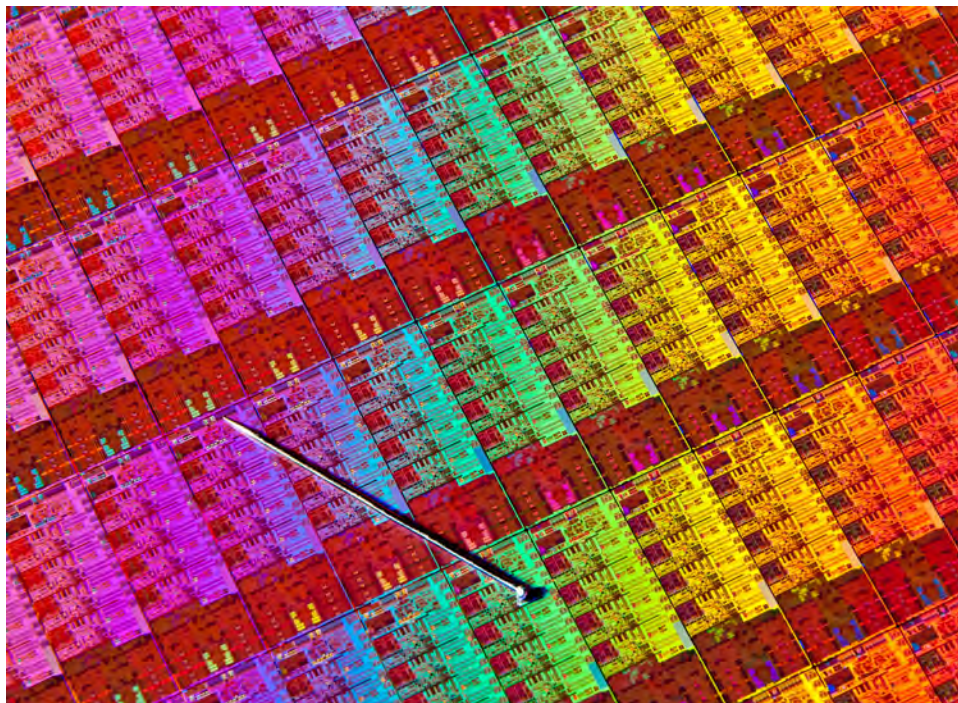
Waterman, et al., *The RISC-V Instruction Set Manual*, UCB/EECS-2011-62, 2011

Instruction Set Architectures (ISAs) are deterministic models

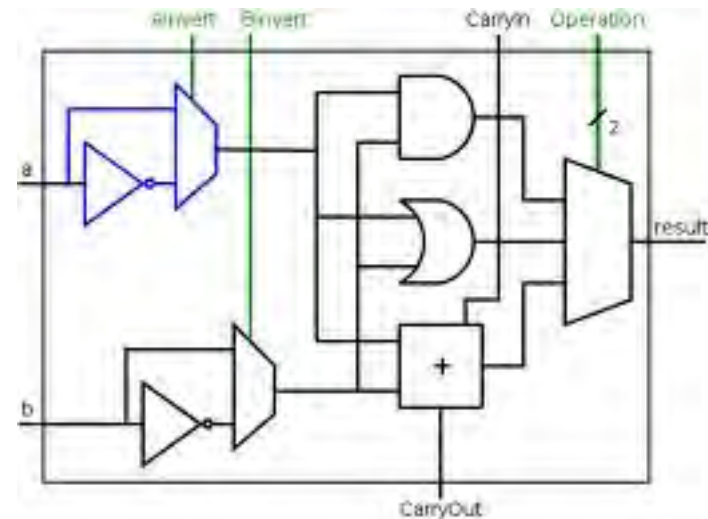


... which relies on yet another deterministic model

Physical System



Model



*Synchronous digital logic
is a deterministic model*



Deterministic Models for the Physical Side of CPS

Physical System

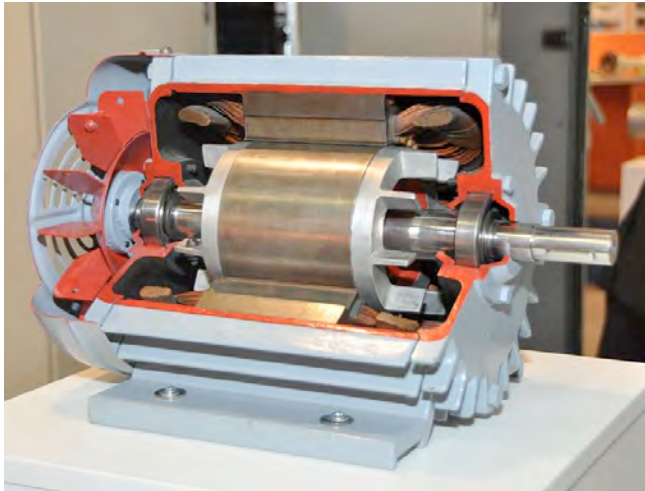


Image: Wikimedia Commons

Model



$$\dot{\mathbf{x}}(t) = \dot{\mathbf{x}}(0) + \frac{1}{M} \int_0^t \mathbf{F}(\tau) d\tau$$

*Differential Equations
are deterministic models*



A major problem for CPS: combinations of deterministic models are nondeterministic



```
1 void initTimer(void) {  
2     SysTickPeriodSet(SysCtlClockGet() / 1000);  
3     SysTickEnable();  
4     SysTickIntEnable();  
5 }  
6 volatile uint timer_count = 0;  
7 void ISR(void) {  
8     if(timer_count != 0) {  
9         timer_count--;  
10    }  
11 }  
12 int main(void) {  
13     SysTickIntRegister(&ISR);  
14     .. // other init  
15     timer_count = 2000;  
16     initTimer();  
17     while(timer_count != 0) {  
18         ... code to run for 2 seconds  
19     }  
20     ... // other code  
21 }
```



$$\dot{\mathbf{x}}(t) = \dot{\mathbf{x}}(0) + \frac{1}{M} \int_0^t \mathbf{F}(\tau) d\tau$$



Timing is not part of software and network semantics

Correct execution of a program in all widely used programming languages, and ***correct delivery*** of a network message in all general-purpose networks has nothing to do with how long it takes to do anything.



Programmers have to step *outside* the programming abstractions to specify timing behavior.

CPS designers have no map!



A Story



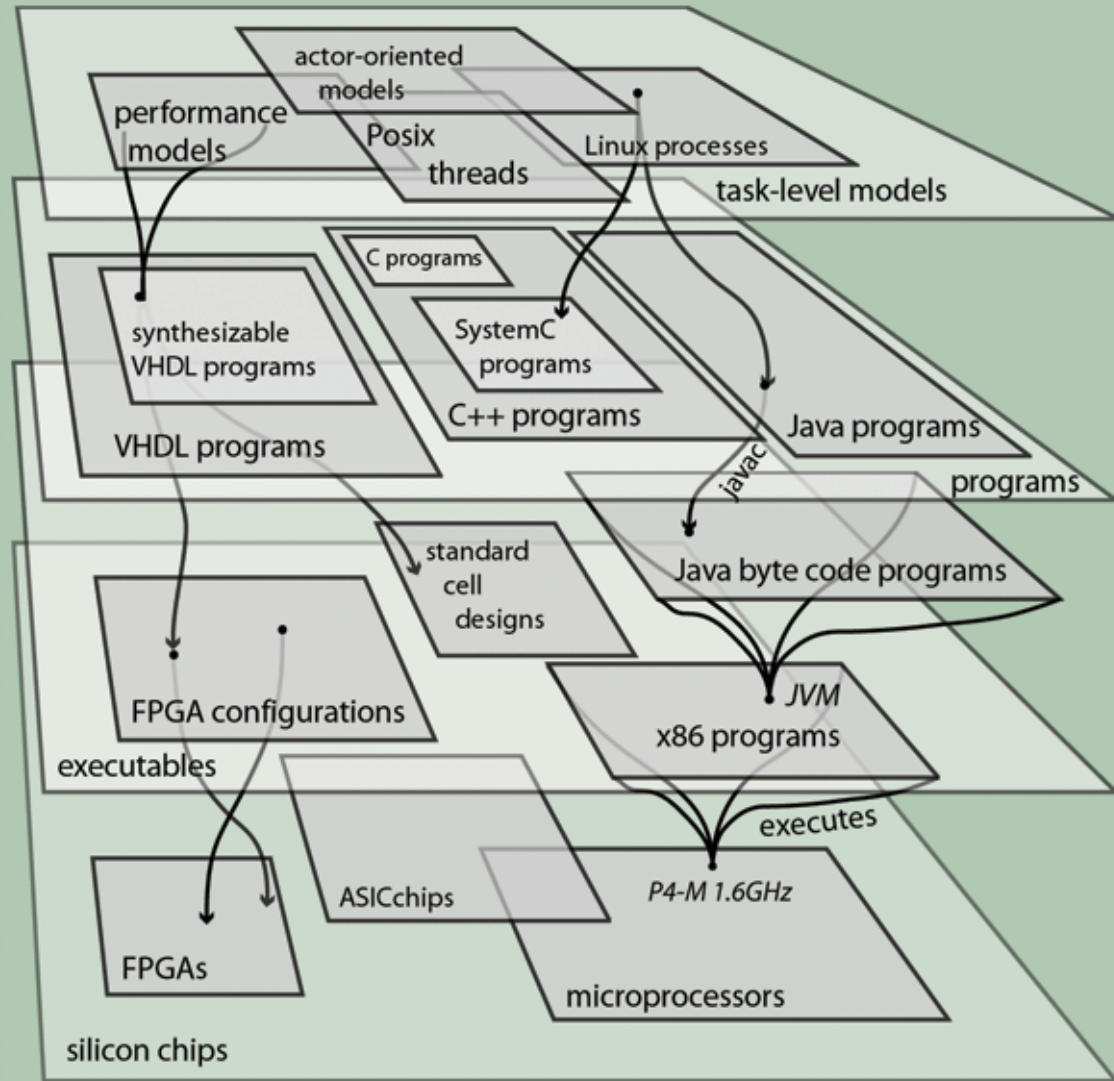
In “fly by wire” aircraft, computers control the plane, mediating pilot commands.





Abstraction Layers

All of which are models except the bottom

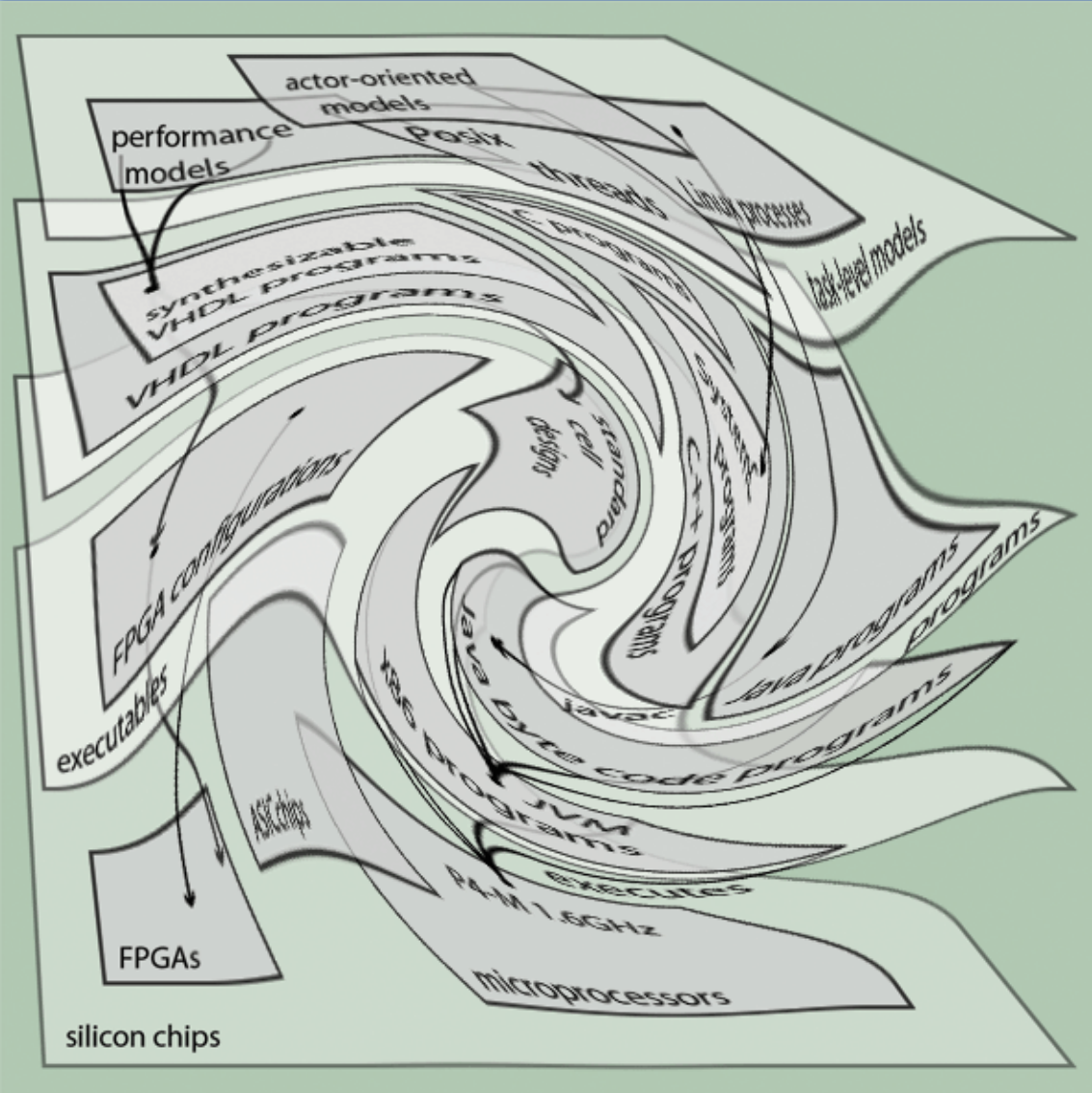


The purpose of an abstraction is to hide details of the implementation below and provide a platform for design from above.



Abstraction Layers

All of which are models except the bottom



Every abstraction layer has failed for the aircraft designer.

The design *is* the implementation.



Determinism? Really?

CPS applications operate in an intrinsically nondeterministic world.

Does it really make sense to insist on deterministic models?



The Value of Models

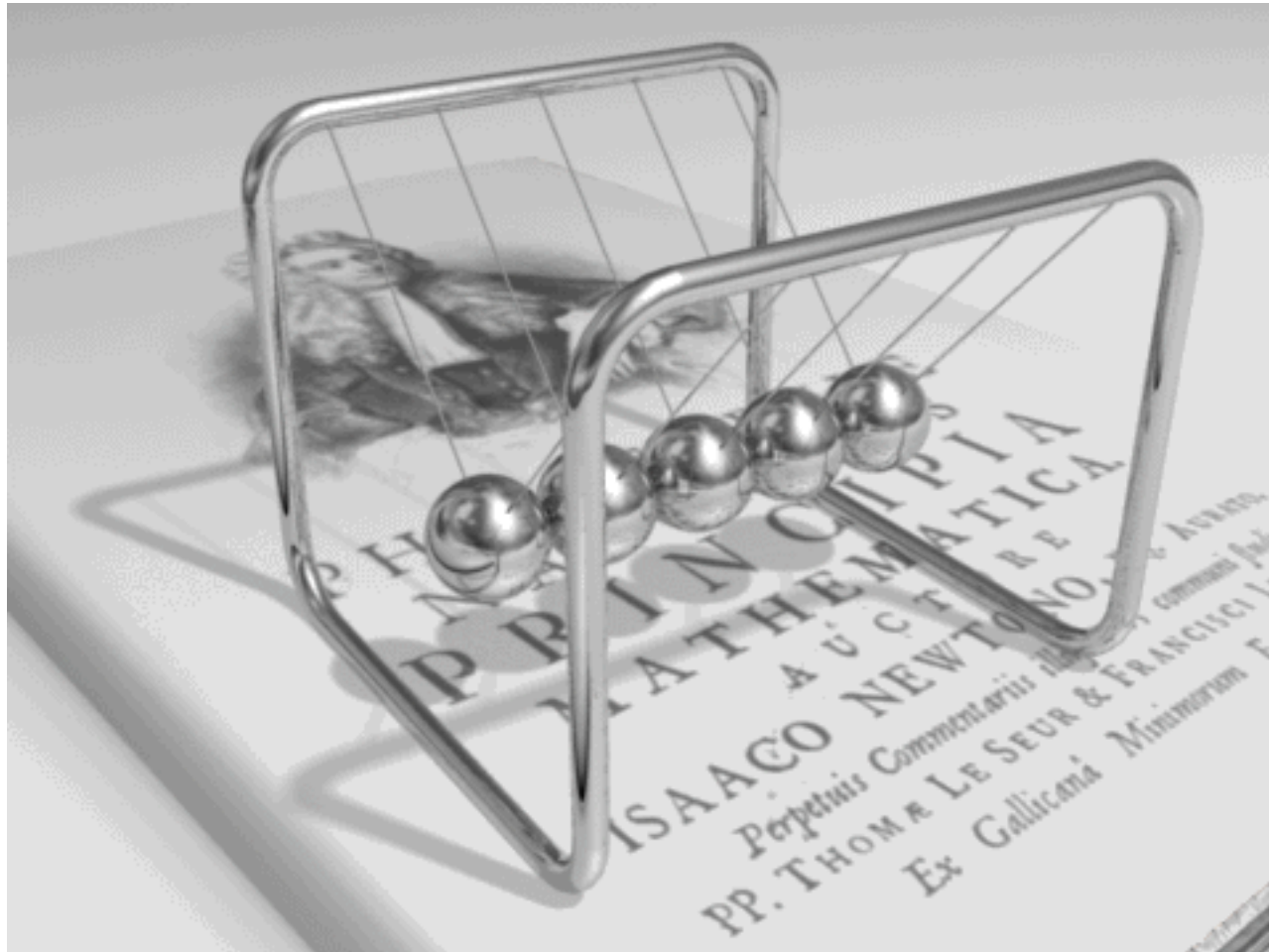
- In *science*, the value of a *model* lies in how well its behavior matches that of the physical system.
- In *engineering*, the value of the *physical system* lies in how well its behavior matches that of the model.

In engineering, model fidelity is a two-way street!

For a model to be useful, it is necessary (but not sufficient) to be able to be able to construct a faithful physical realization.



A Model





A Physical Realization





Model Fidelity

- To a *scientist*, the model is flawed.
- To an *engineer*, the realization is flawed.

I'm an engineer...



For CPS, we need to change the question

The question is *not* whether deterministic models can describe the behavior of cyber-physical systems (with high fidelity).

The question is whether we can build cyber-physical systems whose behavior matches that of a deterministic model (with high probability).



Determinism?

What about resilience? Adaptability?

Deterministic models do not eliminate the need for robust, fault-tolerant designs.

In fact, they *enable* such designs, because they make it much clearer what it means to have a fault!



Existence proofs that useful deterministic models for CPS exist

Deterministic models for CPS with faithful implementations exist:

- PTIDES: distributed real-time software
 - <http://chess.eecs.berkeley.edu/ptides>
- PRET: time-deterministic architectures
 - <http://chess.eecs.berkeley.edu/pret>

These two projects ended in 2015.

Together, these technologies give a programming model for distributed and concurrent real-time systems that is deterministic in the sense of single-threaded imperative programs, and also deterministic w.r.t. to timing of external interactions.



But...

Determinism has its limits.



- Complexity
- Uncertainty
- Chaos
- Incompleteness



Complexity

- Some systems are too complex for deterministic models.
- Nondeterministic abstractions become useful.

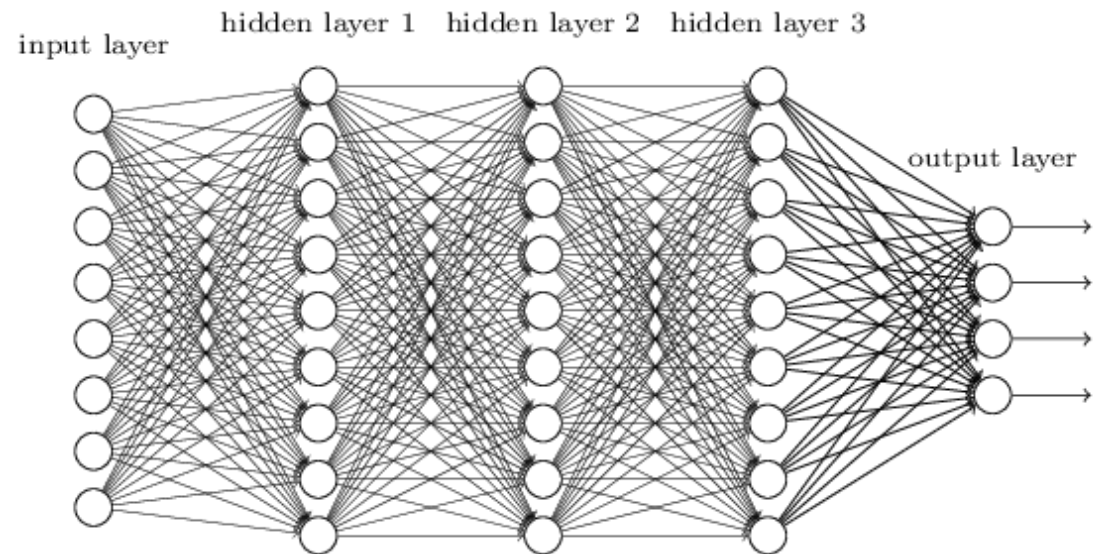


“Iron wing” model of an Airbus A350.



Complexity

- Some systems are too complex for deterministic models.
- Nondeterministic abstractions become useful.



[Deep Learning](http://www.deeplearningbook.org/), draft book in preparation, by Yoshua Bengio, Ian Goodfellow, and Aaron Courville.
<http://www.deeplearningbook.org/>



But...

Determinism has its limits.



- Complexity
- Uncertainty
- Chaos
- Incompleteness



Uncertainty

- We can't construct deterministic models of what we don't know.
- For this, nondeterminism is useful.
- Bayesian probability (which is mostly due to Laplace) quantifies uncertainty.



Portrait of Reverend Thomas Bayes (1701 - 1761) that is probably not actually him.



But...

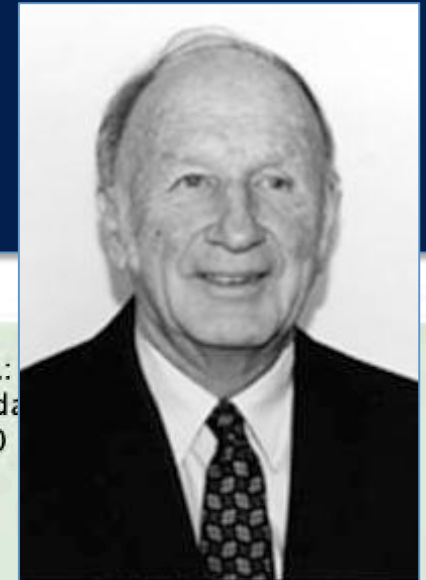
Determinism has its limits.



- Complexity
- Uncertainty
- Chaos
- Incompleteness



Determinism does not imply predictability

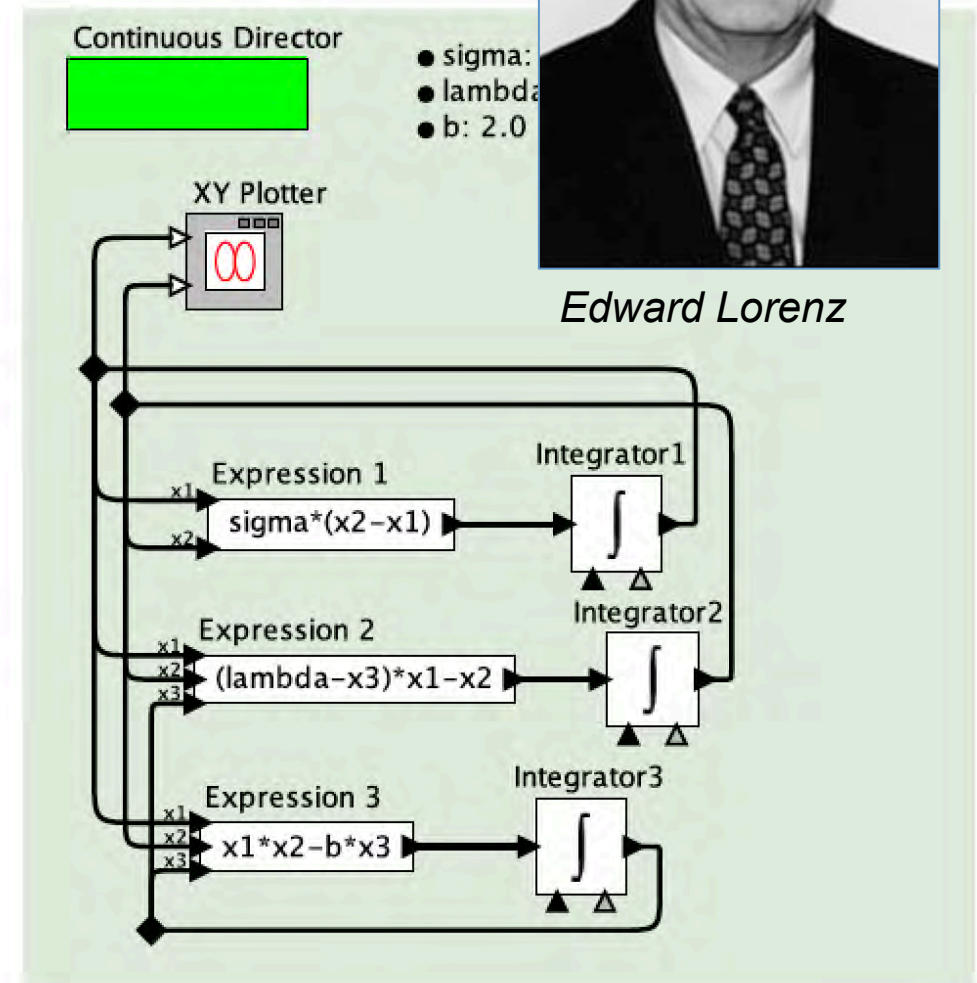


Edward Lorenz

Lorenz attractor:

$$\begin{aligned}\dot{x}_1(t) &= \sigma(x_2(t) - x_1(t)) \\ \dot{x}_2(t) &= (\lambda - x_3(t))x_1(t) - x_2(t) \\ \dot{x}_3(t) &= x_1(t)x_2(t) - bx_3(t)\end{aligned}$$

This is a chaotic system, so arbitrarily small perturbations have arbitrarily large consequences.



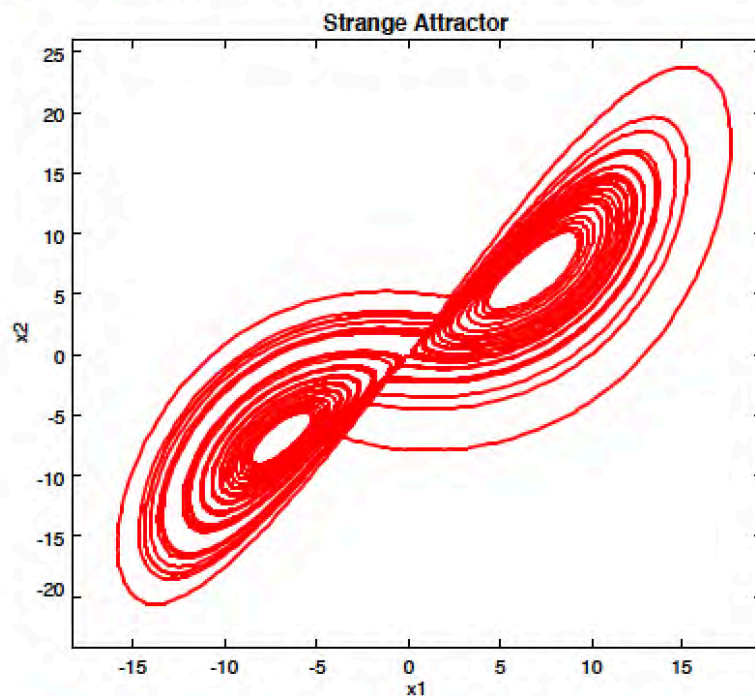


Determinism does not imply predictability

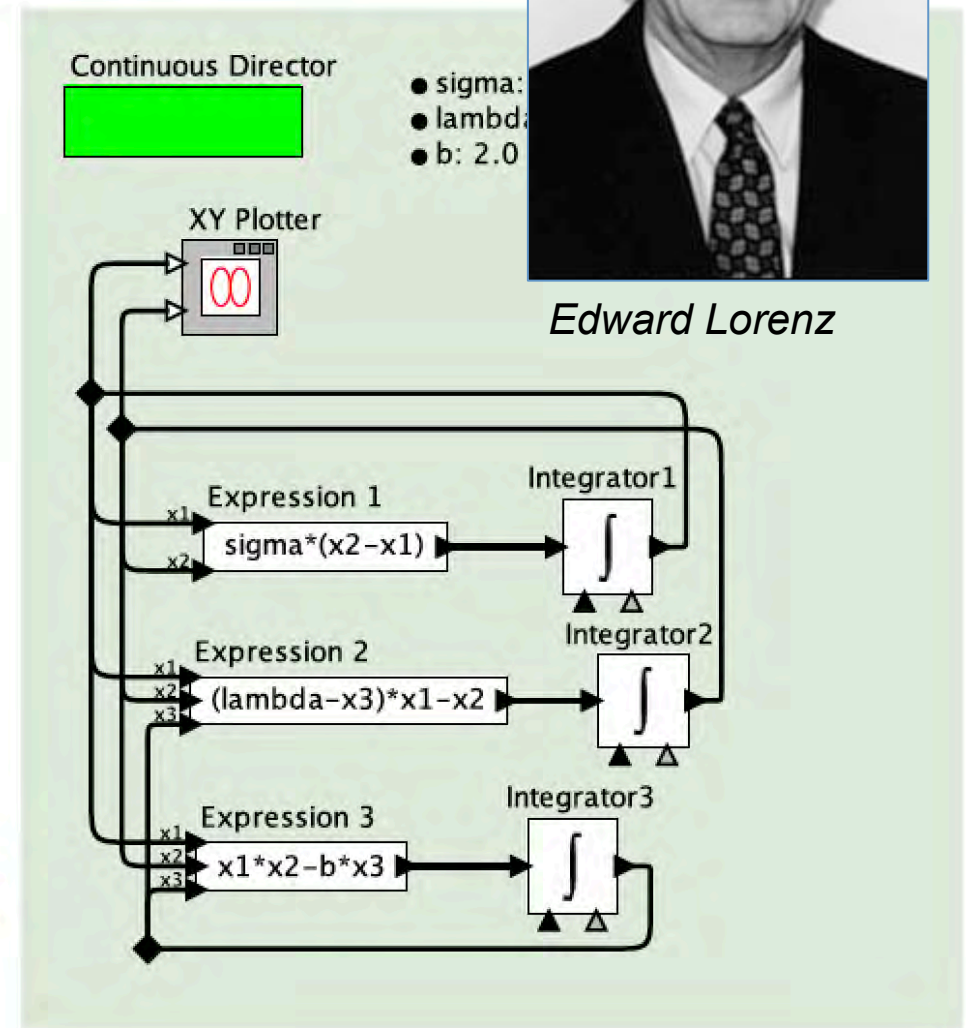


Edward Lorenz

Plot of x_1 vs. x_2 :



The position of a point is not meaningfully predictable even though the system is deterministic.





Determinism does not imply predictability

[Thiele and Kumar, EMSOFT 2015]

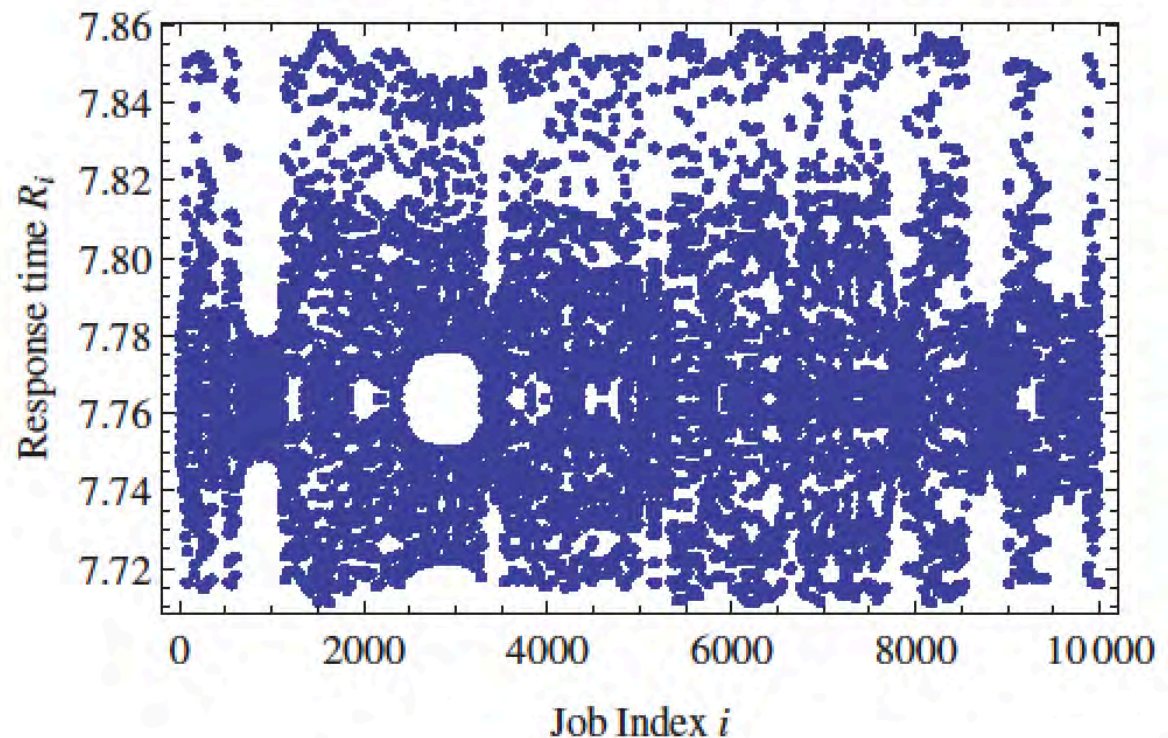


Fig. 15. Response time across jobs for the multi-resource scheduler with $R_s(i-1) = 7.76$ and $R_s(i-2) = 7.74$.



But...

Determinism has its limits.



- Complexity
- Uncertainty
- Chaos
- Incompleteness



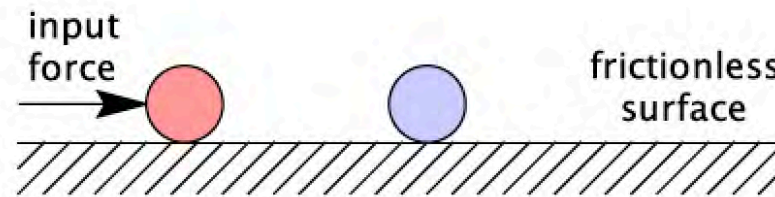
Incompleteness of Determinism

Any set of deterministic models rich enough to encompass Newton's laws plus discrete transitions is incomplete.

[Lee, *Fundamental Limits of Cyber-Physical Systems Modeling*, to appear, ACM Tr. on CPS, 2017]



Illustration of the Incompleteness of Determinism



Conservation of momentum:

$$m_1 v_1' + m_2 v_2' = m_1 v_1 + m_2 v_2.$$

Conservation of kinetic energy:

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

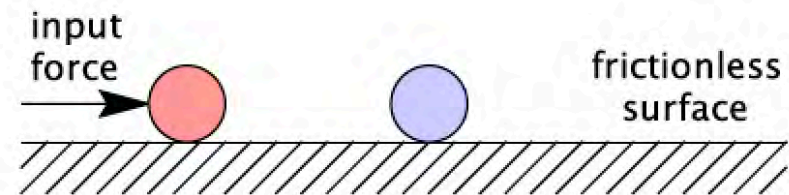
We have two equations and two unknowns, v_1' and v_2' .



Illustration of the Incompleteness of Determinism

Quadratic problem has two solutions.

Solution 1: $v'_1 = v_1$, $v'_2 = v_2$
(ignore collision).



Solution 2:

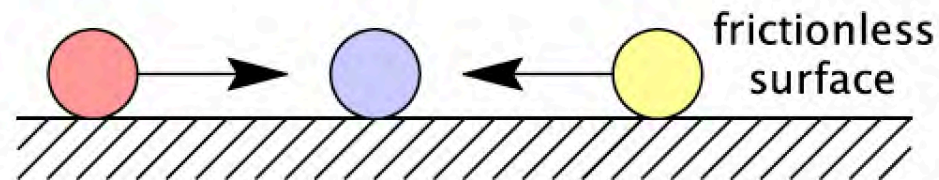
$$v'_1 = \frac{v_1(m_1 - m_2) + 2m_2v_2}{m_1 + m_2}$$
$$v'_2 = \frac{v_2(m_2 - m_1) + 2m_1v_1}{m_1 + m_2}.$$

Note that if $m_1 = m_2$, then the two masses simply exchange velocities (Newton's cradle).



Illustration of the Incompleteness of Determinism

Consider this scenario:

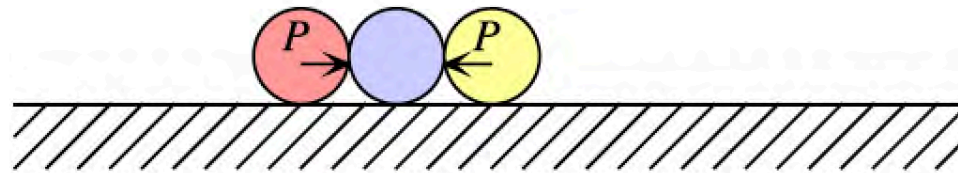


Simultaneous collisions where one collision does not cause the other.



Illustration of the Incompleteness of Determinism

One solution: nondeterministic interleaving of the collisions:

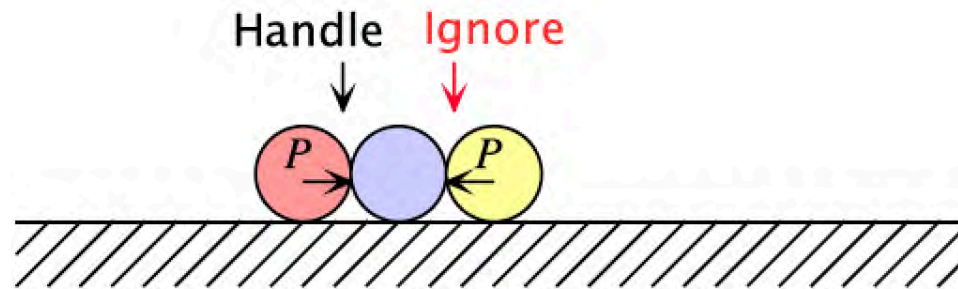


At superdense time $(\tau, 0)$, we have two simultaneous collisions.



Illustration of the Incompleteness of Determinism

One solution: nondeterministic interleaving of the collisions:

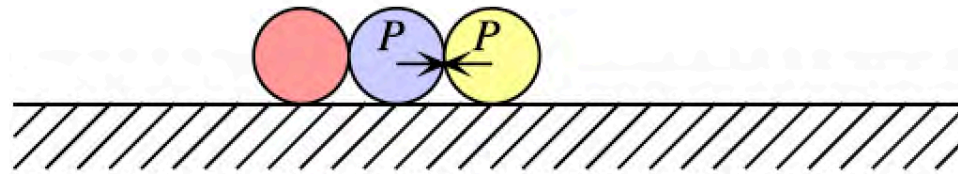


At superdense time $(\tau, 1)$, choose arbitrarily to handle the left collision.



Illustration of the Incompleteness of Determinism

One solution: nondeterministic interleaving of the collisions:

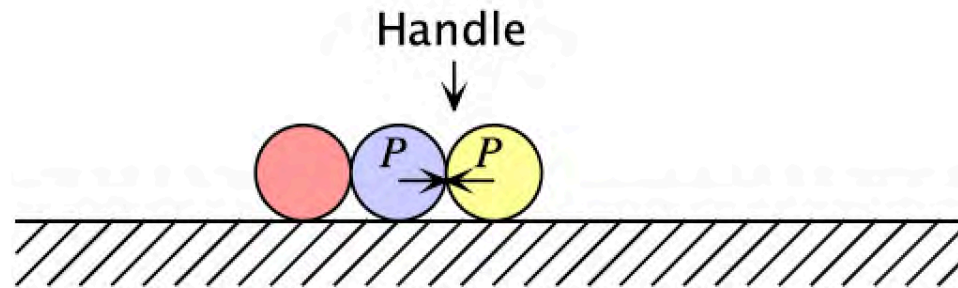


After superdense time $(\tau, 1)$, the momentums are as shown.



Illustration of the Incompleteness of Determinism

One solution: nondeterministic interleaving of the collisions:

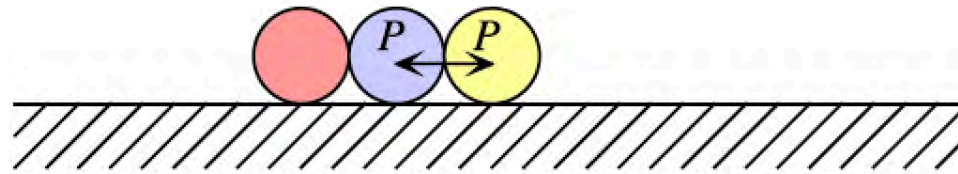


At superdense time $(\tau, 2)$, handle the new collision.



Illustration of the Incompleteness of Determinism

One solution: nondeterministic interleaving of the collisions:

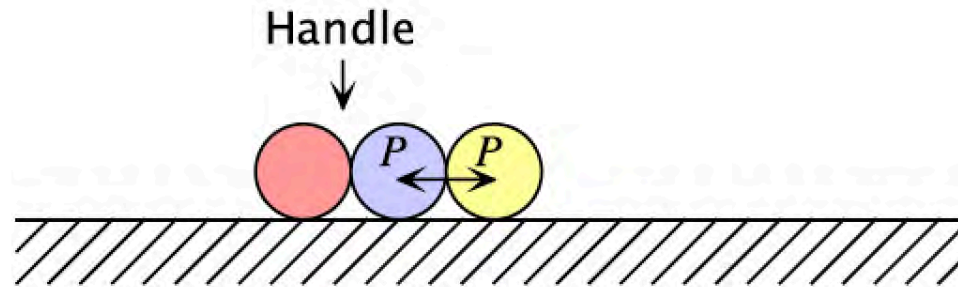


After superdense time $(\tau, 2)$, the momentums are as shown.



Illustration of the Incompleteness of Determinism

One solution: nondeterministic interleaving of the collisions:

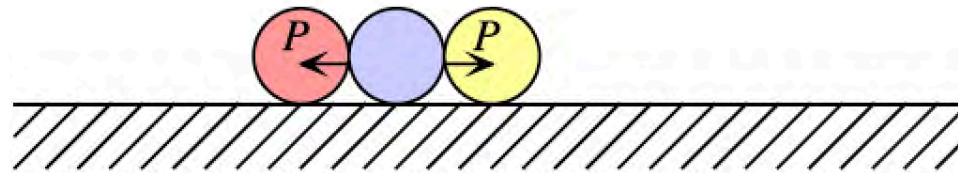


At superdense time $(\tau, 3)$, handle the new collision.



Illustration of the Incompleteness of Determinism

One solution: nondeterministic interleaving of the collisions:

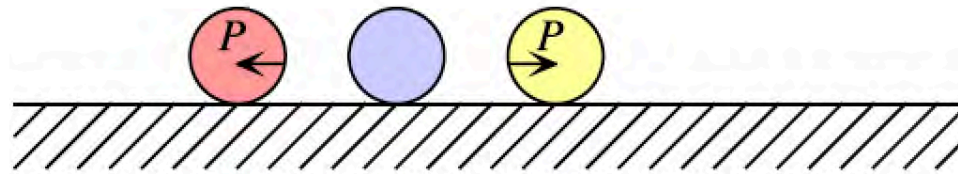


After superdense time $(\tau, 3)$, the momentums are as shown.



Illustration of the Incompleteness of Determinism

One solution: nondeterministic interleaving of the collisions:

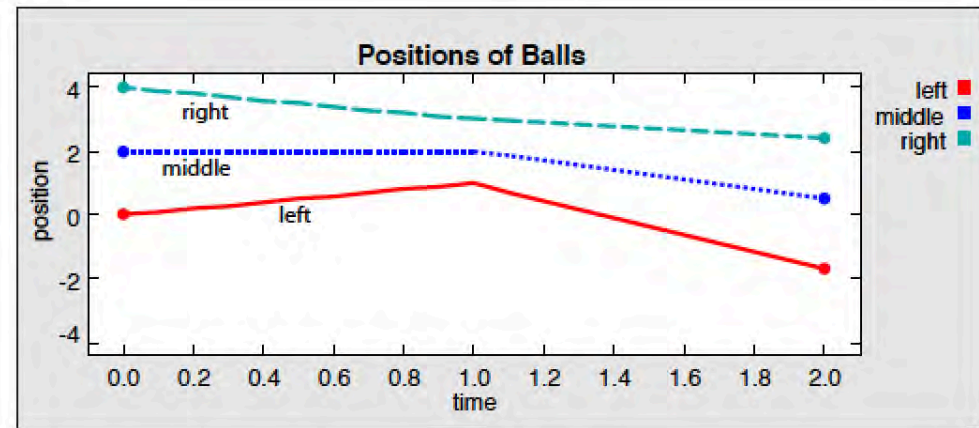


The balls move away at equal speed (if their masses are the same!)

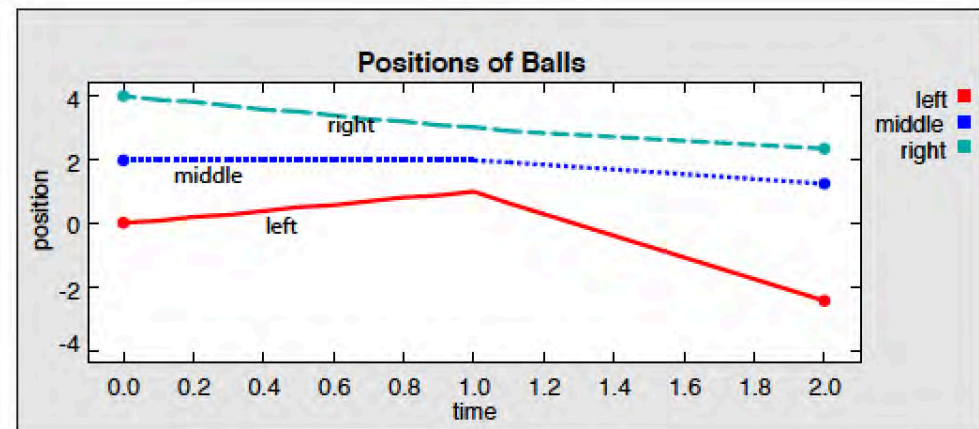


Arbitrary Interleaving Yields Nondeterminism

If the masses are different, the behavior depends on which collision is handled first!



(a)



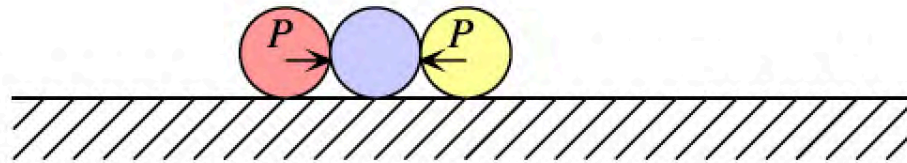
(b)



Recall the Heisenberg Uncertainty Principle

We cannot simultaneously know the position and momentum of an object to arbitrary precision.

But the reaction to these collisions depends on knowing position and momentum precisely.

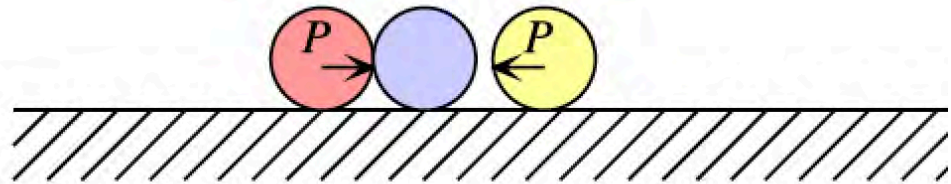




Is Determinism Incomplete?

Let τ be the time between collisions. Consider a sequence of models for $\tau > 0$ where $\tau \rightarrow 0$.

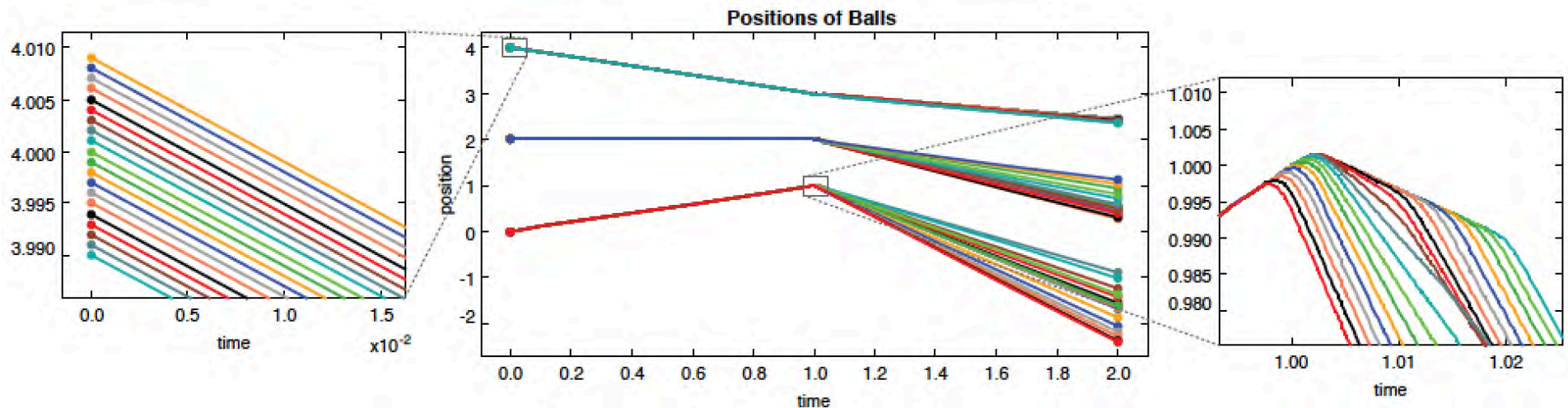
Every model in the sequence is deterministic, but the limit model is not.



- In Lee (2017), I show that this sequence of models is Cauchy, so the space of deterministic models is incomplete (it does not contain its own limit points).
- In Lee (2014), I show that a direct description of this scenario results in a non-constructive model. The nondeterminism arises in making this model constructive.



Rejecting discreteness leads to deterministic chaos



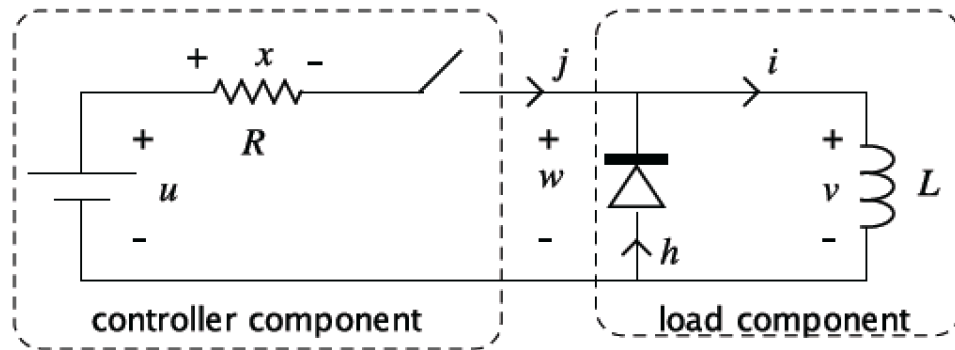
A continuous deterministic model that models the balls as springs is chaotic.



Discrete behaviors cannot be excluded unless we also reject causality

*Example from Lee,
“Constructive Models of Discrete and
Continuous Physical Phenomena,”
IEEE Access, 2014*

A **flyback diode** is a commonly used circuit that prevents arcing when disconnecting an inductive load (like a motor) from a power source.



When the switch goes from closed to open, the causality and direct feedthrough properties of the two components reverse.

There is no logic that can transition from A causes B to B causes A smoothly without passing through non-constructive models.



Summary

- Deterministic models are extremely useful.
- Combining of our best deterministic cyber models and physical models today yields nondeterministic models.
- But deterministic models with faithful implementations exist (in research) for cyber-physical systems.
- Deterministic models aren't always possible or practical due to complexity, unknowns, chaos, and incompleteness.
- Determinism is a powerful modeling tool.
Use it if you can. Back off only when you can't.



Conclusion

Models play a central role in reasoning about and designing engineered systems.

Determinism is a valuable and subtle property of models.

*Forthcoming book
My first for a general audience*

Plato and the Nerd *On Technology and Creativity*

Edward Ashford Lee
MIT Press, 2017

