Introduction to the course: Verifying cyberphysical systems

Verifying cyberphysical systems

August 27th 2019

Sayan Mitra

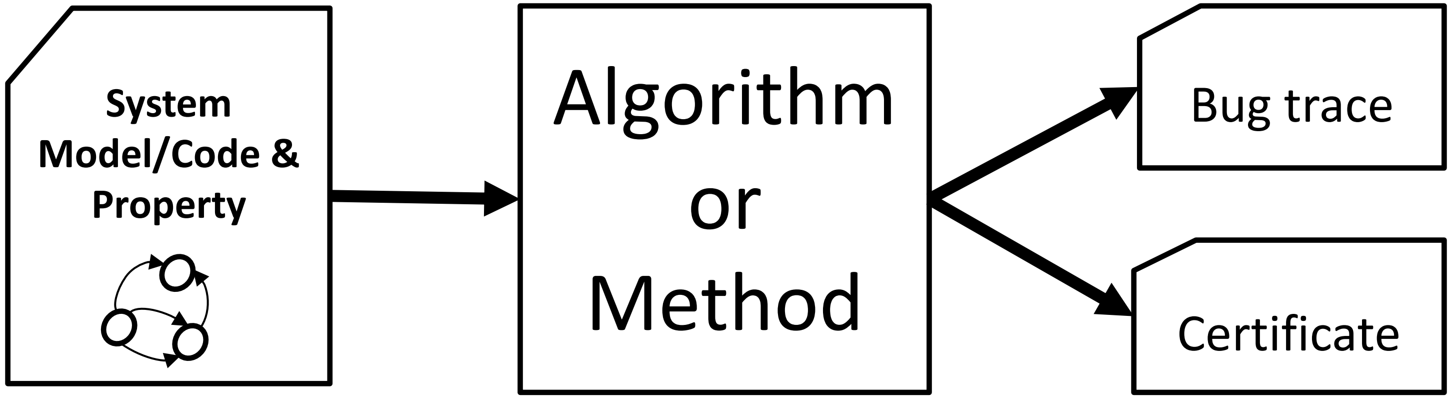
mitras@illinois.edu

Welcome

What is this class about?

INTRODUCTION

The verification problem



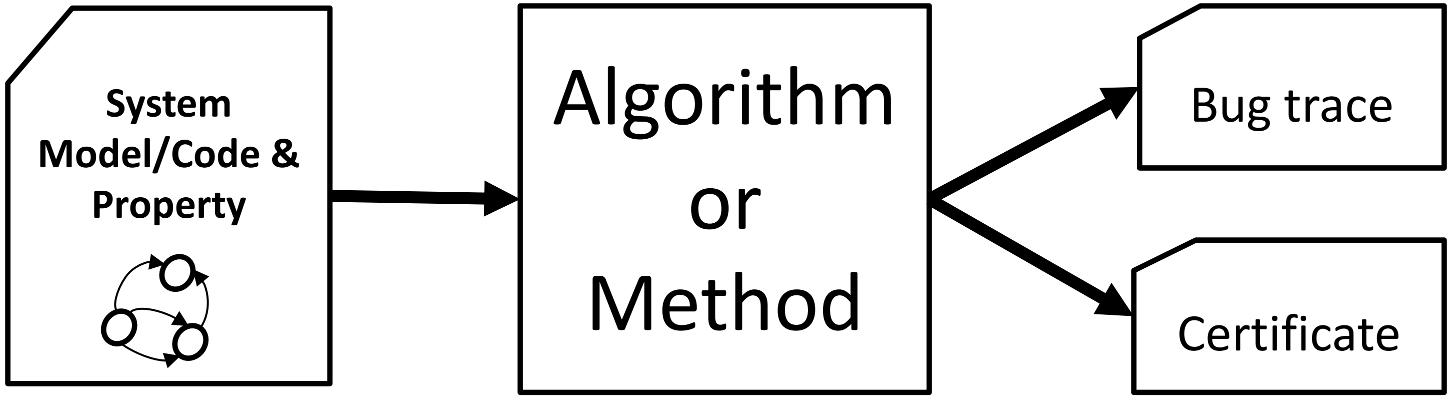
Verification. *The action of demonstrating or proving to be true by means of evidence; formal assertion of truth. (OED)*

An example

System. A subroutine `sort(int a[])` for returning a sorted array of integers in some programming language, e.g. C

A model M for execution of programs in C

Requirement. Output of `sort(int a[])` is the sorted version of the input array `a[]`



counterexample. A particular input array `a` and initialization of `sort` that produces wrong output

A mathematical proof that establishes that `sort(int a[])` works for all inputs in the given model M of C

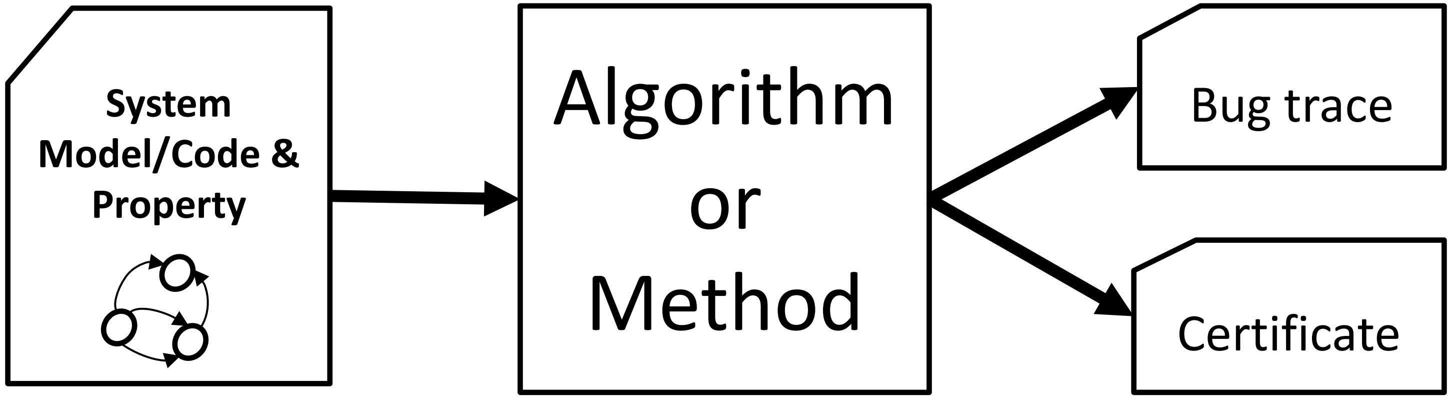
Verifying compiler. Checks that `sort` meets the requirement

An *cyberphysical* example

System. A program/system for *lane keeping control* for vehicles

Model/assumptions for executing such programs including the effects on the physical vehicle

Requirement. The vehicle does not go outside the lane boundaries



Verification tool

counterexample. A particular environment situation (lane geometry, sensor failure, computer configuration) that makes the vehicle go outside lanes

A mathematical proof that establishes that for all *allowed* inputs and environments the vehicle stays with the lane

When can we build such a tool? How expensive is it? How well is it going to work? Under what assumptions?

Verifying cyberphysical system

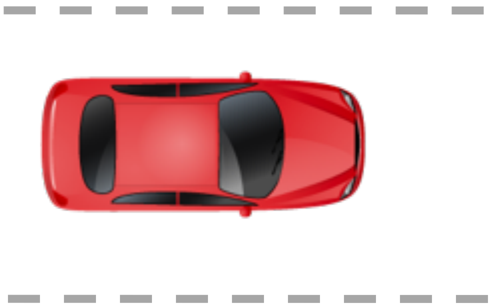
- *Cyberphysical system (CPS)*: a computer controlling something physical. For example, car, drone, medical device, power grid, etc.
- The number of possible behaviors* usually uncountably infinite
- Requirement: Assertions about all *behaviors*
  - Under all nominal conditions the vehicle stays within the lanes
  - Under all nominal driving conditions the emissions are within the prescribed range
  - The drone visits the waypoints while avoiding collisions
  - Insulin pump maintains blood glucose level to within the prescribed range
- Testing: evaluates requirements on a finite number of behaviors
- Verification: aims to prove requirements over all behaviors

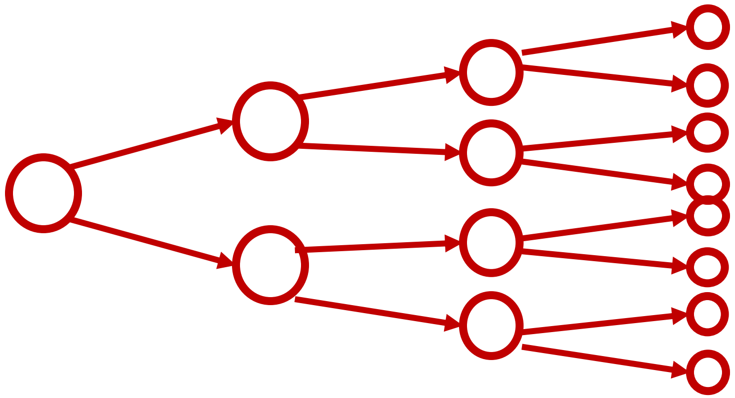
* To be defined precisely

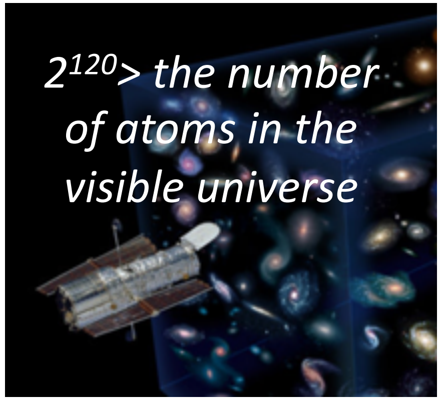
Goal

Write programs that prove correctness of other programs

Suppose *Hamlet's* car has 2 choices every 10ms, how many positions could it be in in 10 seconds? Predicting all futures

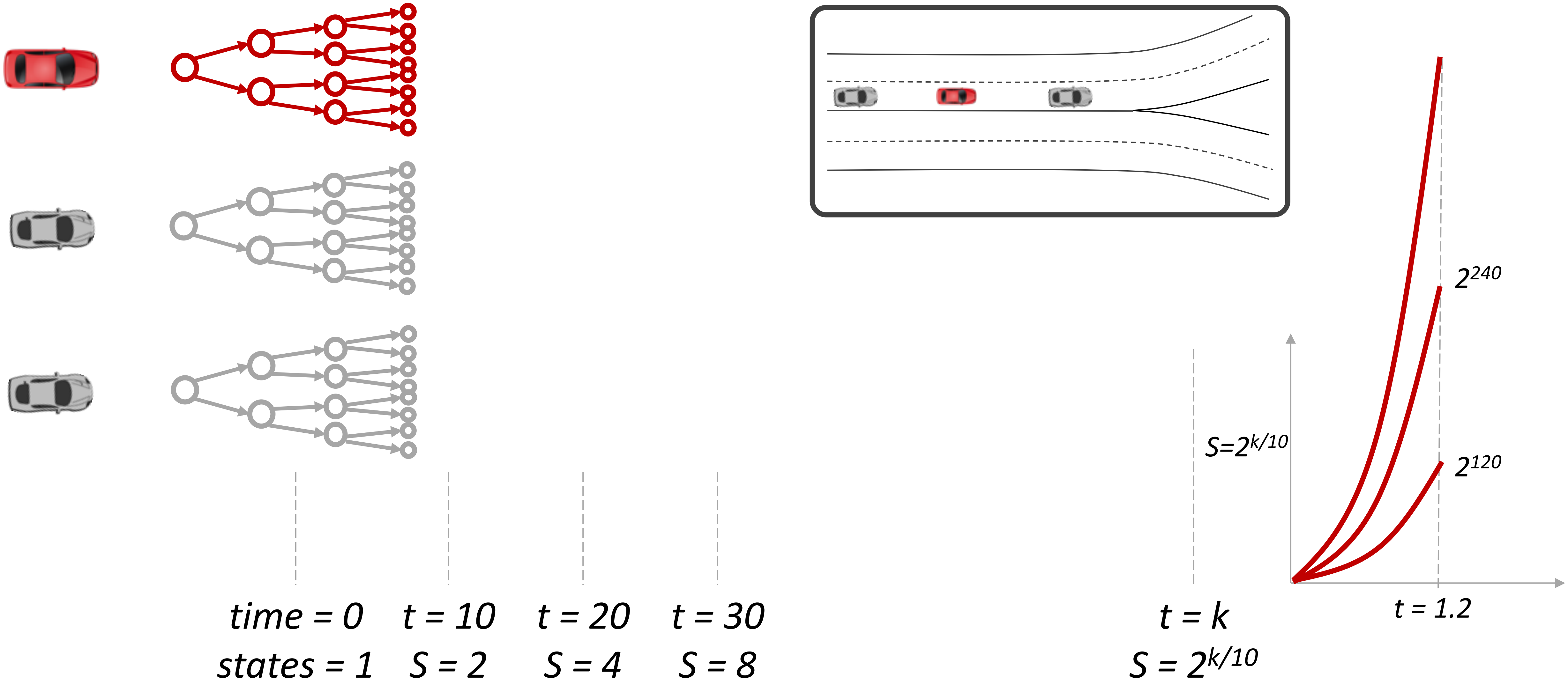






time = 0	*t = 10*	*t = 20*	*t = 30*	*t = k*
states = 1	*S = 2*	*S = 4*	*S = 8*	$S = 2^{k/10}$

State space explosion! Number of states grow exponentially with time!



How many miles must an autonomous car test drive before we call it safe?

200 million miles?

0.07 fatalities per billion passenger miles (commercial flight)

Why is air-travel safe?

Regulations and Audits

What fraction of the cost of developing a new aircraft is in SW?

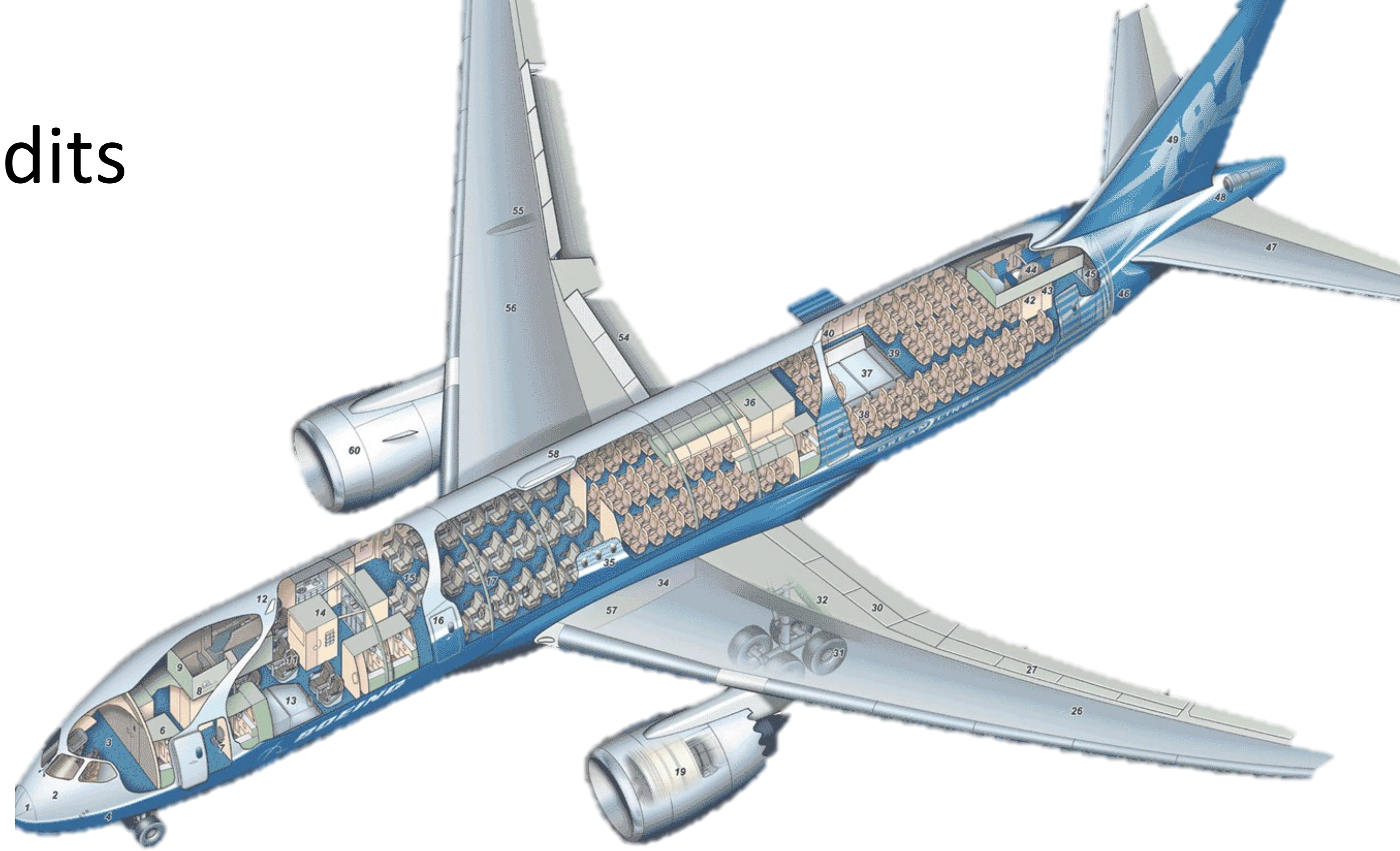
How much does it cost to change 1 line of code?

DO178C

Primary document by which FAA & EASA approves software-based aerospace systems.

DAL establishes the process necessary to demonstrate compliance

Supplement DO-333 supplement of DO-178C identifies aspects of airworthiness certification that pertains to of software using *formal methods*

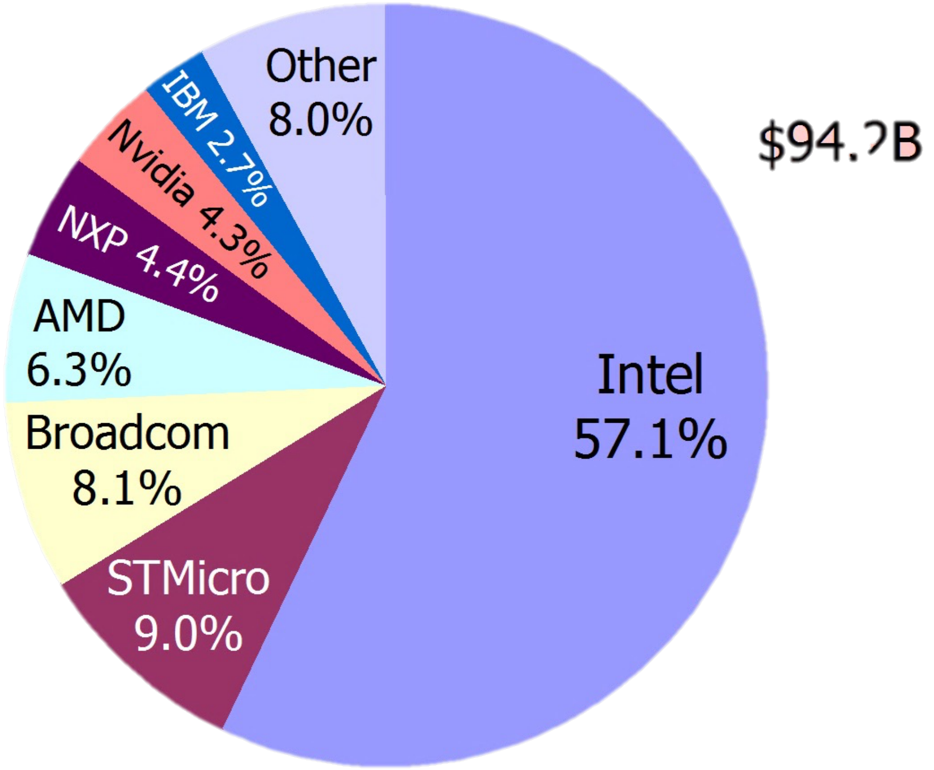


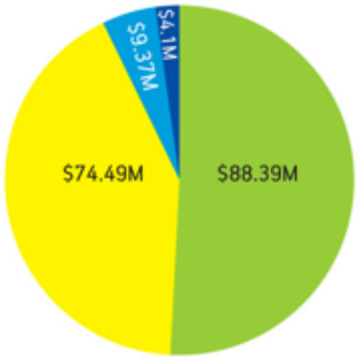
Dev.Assurance Level (DAL)	Hazard Classification	Objectives
A	Catastrophic	71
B	Hazardous	69
C	Major	62
D	Minor	26
E	No Effect	0

Statement Coverage: Every statement of the source code must be covered by a test case

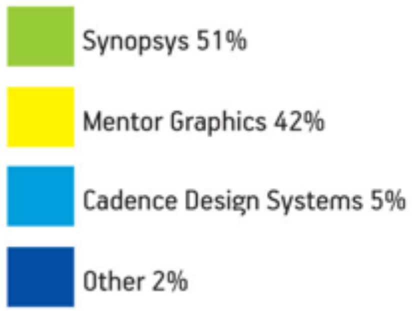
Condition Coverage: Every condition within a branch statement must be covered by a test case

Another earlier success instance: microprocessor industry and supporting design automation tools





Electronic design automation industry



Beyond ECE/CS 584

- Hardware verification (model checking) is now part of engineering practice in the industry
- Automated Device Driver Verification at Microsoft: SLAM tool from MSR; AMAZON Web services verified using TLA
- Formal modeling and analysis is becoming part of certification process for avionics (e.g., ASTREE); adoption for automotive and manufacturing around the corner
- Commercial enterprises
  - Synopsis, Mentor Graphics, Cadence, Coverity, Galois, SRI, etc.
  - More up and coming in the automotive space
- Vibrant, focused research community:
  - Conferences: CAV, TACAS, PLDI, HSCC, EMSoft
  - Faculty and research. positions
  - Turing Awards: Lamport (2014), Clarke, Sifakis & Emerson (2008), Pnueli (1997), Lampson (1992), Milner (1991), Hoare (1980), Dijkstra (1972) …

Verification aims to mathematically prove requirements over all behaviors

- To prove anything, first we have to start with assumptions
- These assumptions will be captures in the *models* of cyberphysical systems

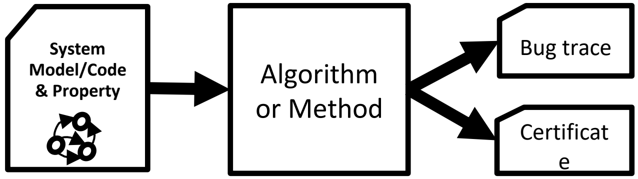
"All models are wrong, some are useful"

- 1/3 of this class is about models
  - Programs, state machines, or differential equations, block diagrams?
  - Discrete or continuous time
  - Discrete or continuous state or both
  - Hybrid, switched
  - Deterministic or nondeterministic or both or neither
  - Composition and interfaces
  - Abstraction
  - Modeling languages, tools

Verifying hybrid models is a very hard problem

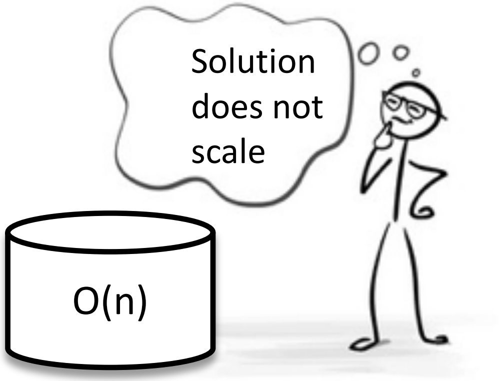
- Verification of hybrid automaton is *undecidable*
  - *No one* can find the is Algorithm of that type

- Approximate and bounded time versions of the problem can be solved algorithmically, but often the algorithms do not *scale* with the size of the model, number of agents, time horizon, etc.

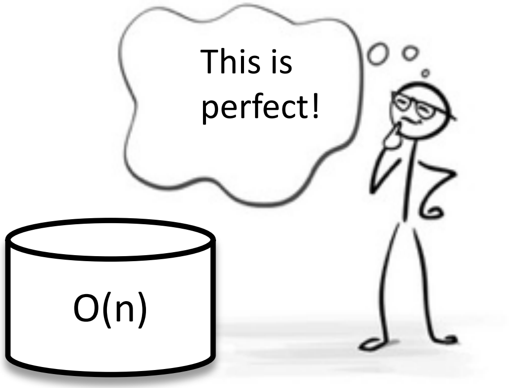


- Models are often hard to get
  - IP protection
  - Too complex, messy
  - Machine learning modules

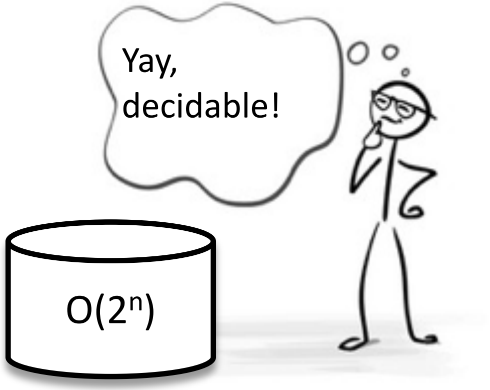
Odd perspectives on scalability



data scientist



algorithmist



verification engineer

Silver linings (course objectives)

- Learn the foundational connections between computer science and control theory
- Model everything
- Introduction to key concepts in formal methods and cyberphysical systems; exposure to some of the most influential ideas in CS and
- Learn powerful algorithms and tools
- Jumpstart research

Invariant, barrier certificates, ranking functions, stability, self-stabilization, convergence, transition system

Programs, state machines, or differential equations, discrete or continuous state or both, Hybrid, switched, Deterministic or nondeterministic or both, composition, interfaces, abstraction, modeling languages, tools

satisfiability modulo theory, semantics, temporal logics, theorem provers, SAF solvers, ranking functions, data-driven verification, HYLAA, C2E2, SpaceEx, Flow*, Z3, …

semester-long project, feedback, presentation, hardware, software, and data resources

How the course works

ADMINISTRIVIA

Illinois 2019 Edition

- https://wiki.illinois.edu/wiki/pages/viewpage.action?pageId=642598908