Programs and Proofs

KC Sivaramakrishnan

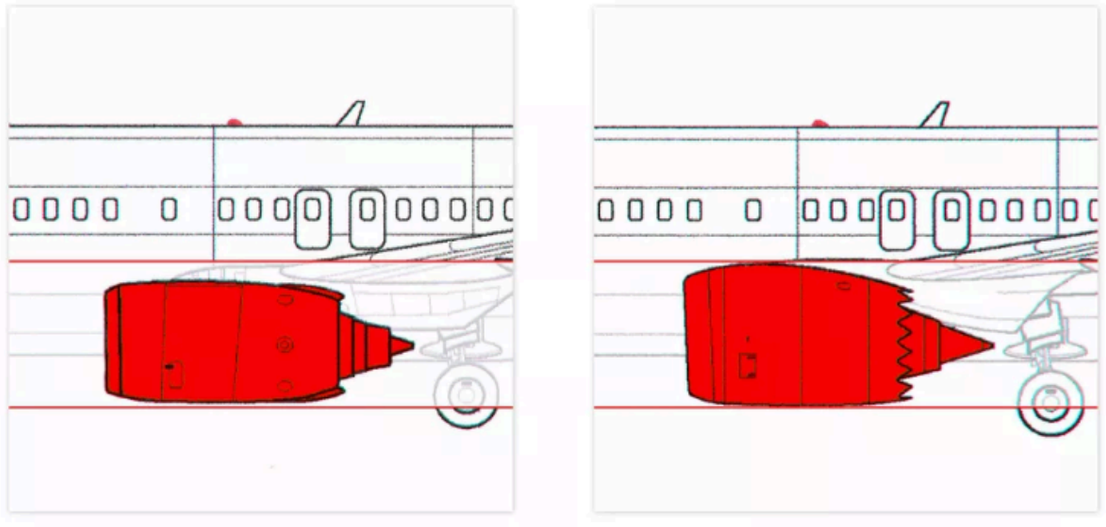
Spring 2020

Building Reliable Software

- Suppose you run a software company
- Support you've sunk 30+ person-years into developing the "next big thing":
  - ★ Boeing Dreamliner2 flight controller
  - ★ Autonomous vehicle control software for Tesla
  - ★ Gene therapy DNA tailoring algorithms
  - ★ Super-efficient green-energy power grid controller
- How do you avoid disasters?
  - ★ Turns out software endangers lives

Boeing 737 Max Crashes

- Involved in two crashes
  - ✦ Lion Air Flight 610 on October 29, 2018 — 189 dead
  - ✦ Ethiopian Airlines Flight 302 on March 10, 2019 — 157 dead
- The crash is attributed to design errors including flight control software
  - ✦ The position of larger engines on 737 Max generated addition lift



Engine placement on the third-generation 737 NG (left) versus the MAX (right).

Boeing 737 Max Crashes

- Manoeuvring Characteristics Augmentation System (MCAS)
  - ✦ Software to sense angle of attack (AoA) from a sensor and automatically compensate
- Crashes due to AoA sensor data but also due to MCAS software
- Every time MCAS was switched on and off again, it acted like first time pitching nose lower
  - ✦ incorrect spec not including history
- Max 0.8 degrees pitch during testing, which was changed to 2.4 after
  - ✦ Executing conditions not reflective of testing
- MCAS completely ignored that pilots were desperately pulling back on the yoke
  - ✦ Incorrect spec not considering environment

Not an isolated incident

- NASA's Mars Climate Orbiter
  - ✦ A sub contractor on the engineering team failed to make a simple conversion from English units to metric
  - ✦ $125 million
- Ariane 5 Flight 501
  - ✦ The software had tried to cram a 64-bit number into a 16-bit space.
  - ✦ Crashed both the primary and the backup computer
  - ✦ $500 million payload lost + $XXX to fix the flaw.
- Hawaii Sends Out a State-Wide False Alarm About a Missile Strike
  - ✦ there were "troubling" design flaws in the Hawaii Emergency Management Agency's alert origination software.
- The Equifax social security hack
  - ✦ 143 million of their consumer records (names, SSN, credit card numbers) were stolen by attackers.

Approaches to Validation

- Social
  - ✦ Code reviews
  - ✦ Extreme/pair programming
- Methodological
  - ✦ Design patterns
  - ✦ Test-driven development
  - ✦ Version control
  - ✦ Bug Tracking
- Technological
  - ✦ Static analysis
  - ✦ Fuzzers
- Mathematical
  - ✦ Sound Type Systems
  - ✦ Formal verification

Less formal: Techniques may miss problems in programs

All of these methods should be used!

Even the most formal can still have holes:

- did you prove the right thing?
- do your assumptions match reality?

More formal: eliminate *with certainty* as many problems as possible.

Verification

- Scaled to 10s of lines of code in 1970s
- Now, research projects scale to real software:
  - ✦ CompCert: A verified C compiler
  - ✦ seL4: verified microkernel OS
  - ✦ Ynot: verified DBMS, web services
- In another 40 years?

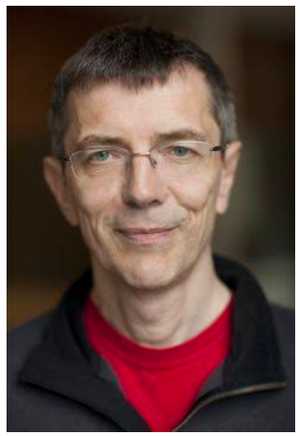
Proof Assistants

- You give assistant a theorem
- You and assistant cooperate to find the proof
  - ✦ Human guides the construction
  - ✦ Machine does the low-level details
- Example: Coq, NuPRL, Isabelle HOL

Coq

- 1984: ***Coquand*** and ***Huet*** implement Coq based on *calculus of inductive constructions*
- 1992: Coq ported to Caml
- Now implemented in OCaml

Thierry Coquand



Gérard Huet

Automated Theorem Proving

- You give the prover a theorem
- The prover either:
  - ✦ Finds a proof
  - ✦ Finds a counter example
  - ✦ Times out
- Eg,
  - ✦ Z3: Microsoft has started shipping with device driver developer kit since Windows 7
  - ✦ ACL2: used to verify AMD chip compliance with IEEE floating point specification, as well as parts of the Java virtual machine

F*

- A solver-aided (Z3) general purpose programming language
- Write programs and write theorems about the programs
  - ✦ F* will discharge the proof obligations to the Z3 solver, but proofs can also be interactive
- Programs can be extracted to OCaml, F#, C, WASM and ASM.
- Main use case is Project Everest at Microsoft — a drop in replacement for HTTPS stack
  - ✦ Verified implementations of TLS 1.2 and 1.3, and underlying cryptographic primitives.

This course

- *Providing a mathematical foundation for rigorous analysis of realistic software systems*

  ✦ Increasingly on demand as almost everything humans interact with is increasingly mediated by software

- We will look at

  ✦ Formal logical reasoning about **program correctness** through

  ✦ **Coq proof assistant**, a tool for machine checked mathematical theorem proving and

  ✦ **F***, a general-purpose programming language aimed at program verifi cation

Why Proof Assistants / Solver-aided PLs?

- Reasoning about program correctness presupposes the ability to read and write mathematical proofs
  - ✦ Humans are bad at writing proofs with pen-and-paper — terribly buggy!
- Proof assistants allow humans to carefully construct machine checked proofs
  - ✦ “obvious to see that it holds” is no longer possible
- Proof assistants = 1 TA per student!
- Homework
  - ✦ Watch “Lambda: the Ultimate TA” by Benjamin Pierce
    - ✤ https://vimeo.com/6615365

Course Contents

- Basics of mathematical logic
  - ✦ Logic::CS = Calculus::EE,Civil,Mech
- Functional Programming
  - ✦ Programs as data, polymorphism, recursion
  - ✦ Specification and verification
- PL theory
  - ✦ transition systems, operational semantics, lambda calculus, Hoare logic, separation logic, weakest precondition, dependent types, monadic effects, etc.

Course Details

- Lectures will be mostly developing programs and proofs interactively
  - ✦ In Coq and F-star
  - ✦ Students are encouraged to bring their laptops and follow along.
- CS3100 OCaml portions are a pre-requisite
  - ✦ Please go through the lecture materials (available on my website) if you aren't comfortable with functional programming.
- Weekly assignments
  - ✦ Expect them to consume 8-10 hours (but may take significantly longer/shorter).
- Collaboration encouraged but not plagiarism.
  - ✦ For example, OK to discuss intermediate lemma, but no copying of proof is allowed.
  - ✦ Will follow the institute policy on plagiarism

Course Details

- Grading: 60% assignments, 20% mid term, 20% final exam
- Office hours
  - ✦ You will need significant assistance with Coq / F*
  - ✦ My calendar is at http://kcsrk.info/calendar
  - ✦ Please drop by my office / fix up a time by sending email to kcsrk@iitm.ac.in
- Exams will also be lab based
  - ✦ Details to be worked out later.
- See the course website http://kcsrk.info/cs6225_s20_iitm for topics and announcements
- Finally, offering this course for the first time
  - ✦ Would like to get continual and honest feedback
  - ✦ This is not an easy course, but hopefully should be quite fun!

Textbooks

- For Coq, we will be following
  - ✦ Adam Chlipala, **Formal Reasoning about Programs**
  - ✦ Freely available here: http://adam.chlipala.net/frap/
- For F*, there is no recommended text
  - ✦ We will be basing our lectures on the F* talks and tutorials available on the F* website: https://www.fstar-lang.org/

Fin!