CS 292C Computer-Aided Reasoning for Software

Lecture 1: Introduction

Yu Feng

Fall 2020

Outline of this lecture

- Introducing the cast
- Motivation and goals
- Course structure

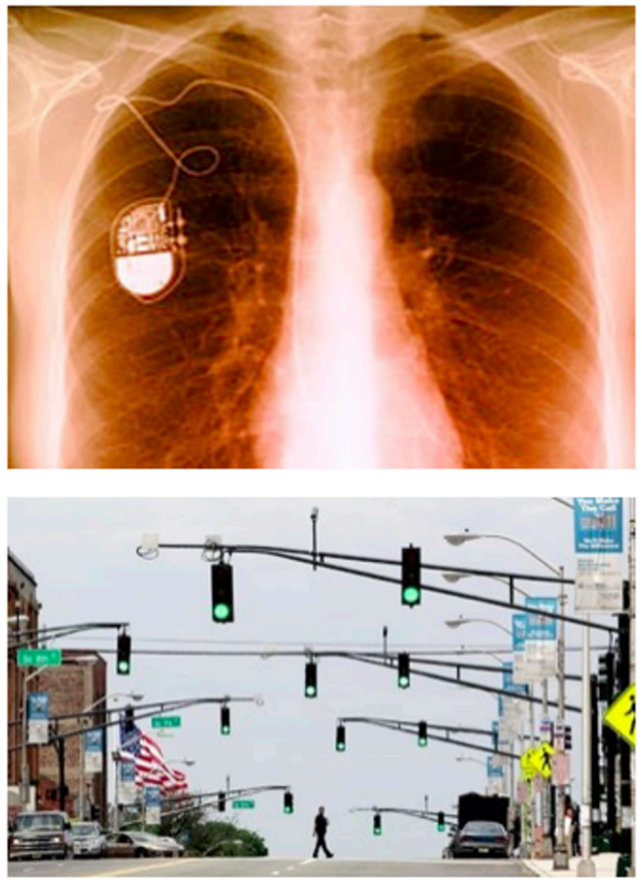
Introducing the cast

Y’all are playing the lead, not audience!

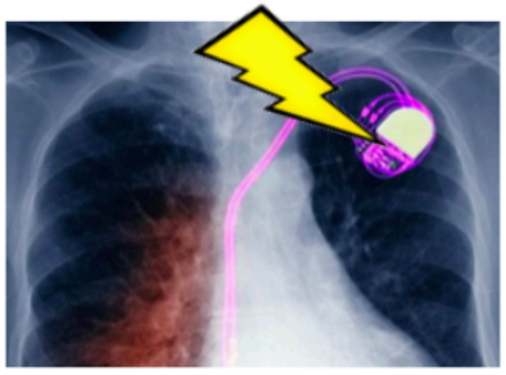
Introducing the cast

- Instructor: Yu Feng
- Email: yufeng@cs
- Office hour: Fri 9am
- Virtual office: https://ucsb.zoom.us/my/yufeng

Software is great



Software is NOT so great



How to make a robust bridge

Make it thicker!

How to make robust software

Testing

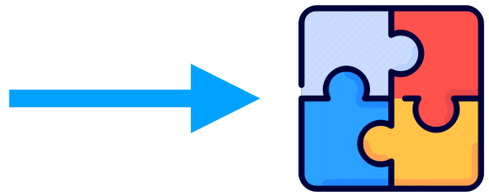
Screening

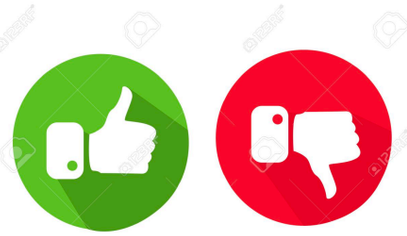
It is undecidable!

Goals of this course

ϕ **(Will this app stole my credit card?)**

Specification





Program verification

ϕ **(Make sure this app won't stole my credit card)**

Specification

Program synthesis

Goals of this course

Open the blackbox

Satisfiability Modulo Theories Solver (SMT solver)

Course structure: prerequisites

- Discrete math
- Compilers
- Programming languages

Course structure: logistics

Website: https://github.com/fredfeng/CS292C

Q&A: https://github.com/fredfeng/CS292C/issues

You need a Git account to post questions!

Workload: medium

- 3 programming assignments
- Paper reviews & presentation
- Final project

Course structure: syllabus

Date	Topic	Slides	Read	Out	Due
10/5	Introduction	[lec1]			
10/7	Solver-Aided Programming I (Rosette)	[lec2]	R1	HW1	
10/12	Solver-Aided Programming II (Neo)	[lec3]	R2		
10/14	SAT Solving Basics	[lec4]			R1
10/19	A Modern SAT Solver	[lec5]			
10/21	Applications of SAT	[lec6]	R3		HW1,R2
10/26	SAT Modulo Theories	[lec7]			
10/28	Combining Theories	[lec8]		[HW2]	R3
11/2	The DPLL(T) Framework	[lec9]			Proposal
11/4	Reasoning about Programs using Hoare logic I	[lec10]	R4		
11/9	Reasoning about Programs using Hoare logic II	[lec11]			

Course structure: grading

- Programming assignments: 15% (3 programming assignments, 5% each)
- Paper reviews: 20% (4 papers, 5% each)
- Paper presentation: 10%
- Final Project: 50%
    - Team formed by deadline: 5%
    - Project proposal: 15%
    - Project presentation: 15%
    - Final report: 15%
- Class Participation: 5%

Course structure: project

- Types of final projects
  - Re-implement/Extend a system from a paper
  - Apply verification/synthesis techniques to your favorite domain
  - …
- Criteria
  - Quality of execution
  - Originality
  - Scope
  - Related to program analysis, verification, or synthesis

Example projects

- Attack synthesis for safety-critical compilers
- Synthesize smart contracts from Temporal Specs
- Data visualization from natural languages
- Exploit generation for Android apps
- Verifying fairness in smart contract
- Vulnerabilities repairing using “big-code”
- …

TODOs for this week

- Create your Git account
- Install Rosette and Neo
  - Install Rosette: https://docs.racket-lang.org/rosette-guide/ch_getting-started.html
  - Install Neo: https://github.com/fredfeng/Trinity
- Start to look for partners for your final project!