CS 190I Program Synthesis for the Masses

Lecture 1: Introduction

Yu Feng

Spring 2021

Outline of this lecture

- Introducing the cast
- Motivation and goals
- Course structure

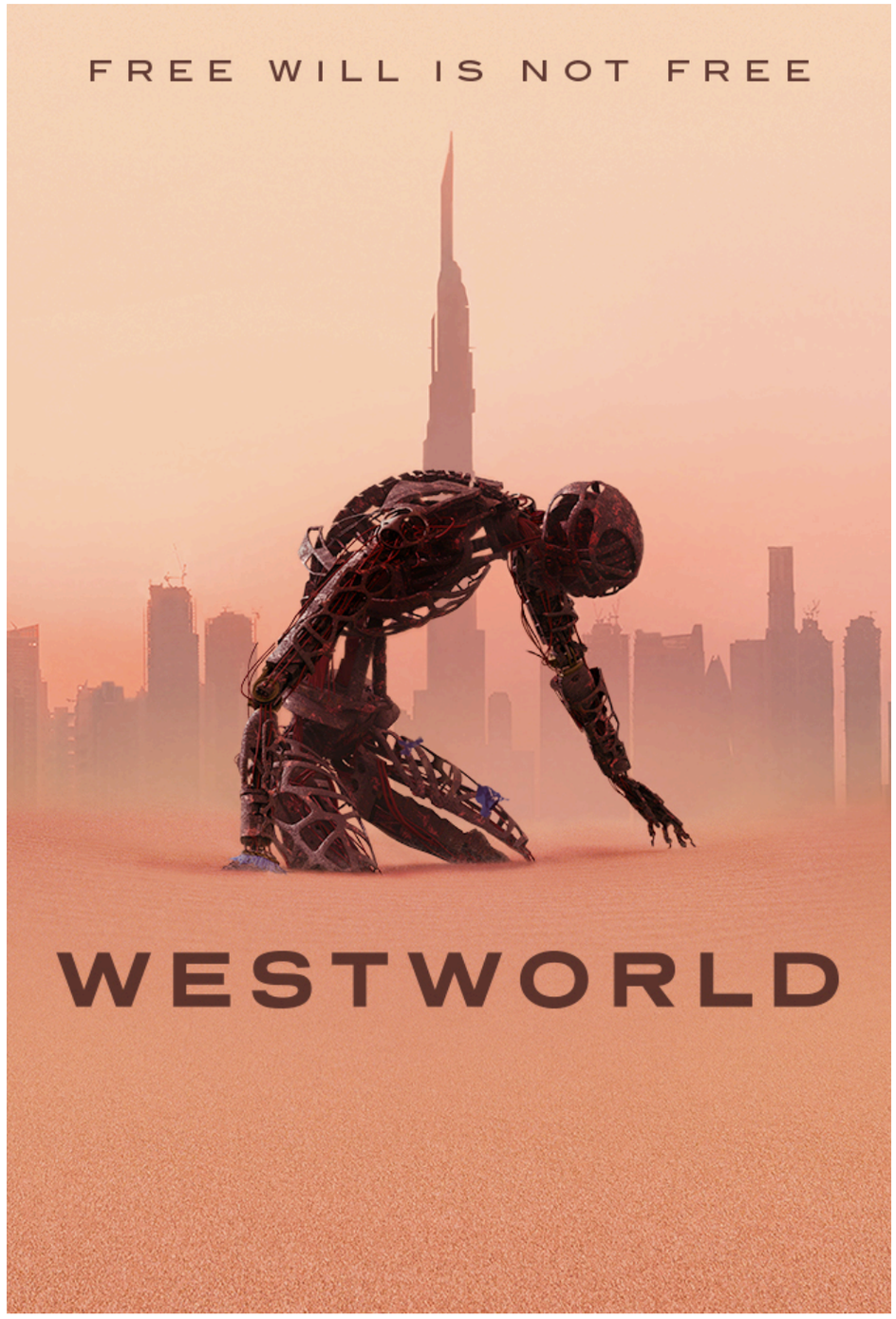
Introducing the cast

- Instructor: Yu Feng
- Research areas: programming languages, program verification, program synthesis, data science, and security
- Email: yufeng@cs.ucsb.edu
- Office hour: Fri 9am
- Virtual office: the same link as the current one

Introducing the cast

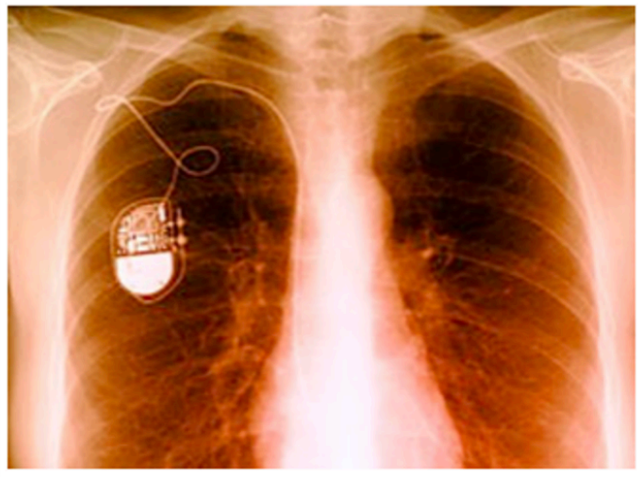
- TA: Yanju Chen
- Email: yanju@cs.ucsb.edu
- Office hour: 4pm-5pm, Monday
- Virtual office: https://ucsb.zoom.us/my/yanju

Ultimate goal of CS

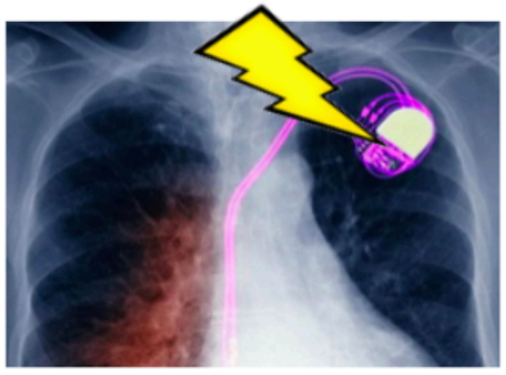


Automatically
Generating Programs!

Software is great



Software is NOT so great



How to make a robust bridge

Make it thicker!

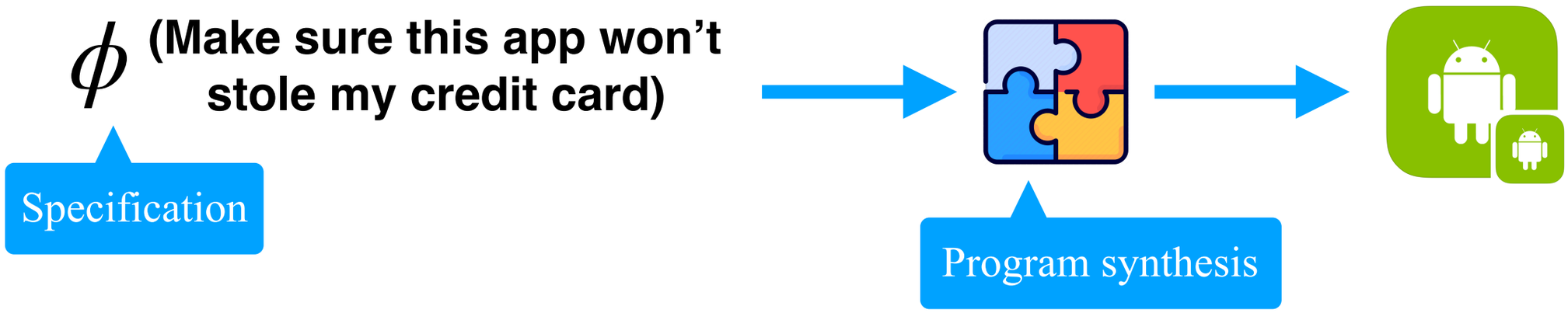
How to make robust software

Testing

Screening

It is undecidable!

Program synthesis



Program Synthesis aims to generate a program from a collection of artifacts that establish semantic and syntactic requirements (i.e., specifications)

Applications using Synthesis

Systems

SOSP'19, OSDI'18,
SOSP'17, OSDI'16, OSDI'20

Blockchain

ASE'20

Browser engines

PPoPP'13

Biology

POPL'14

Education

Data science

PLDI'18, PLDI'17

Healthcare

CAV'16

HPC

ASPLOS'16, OSDI'18

Gaming

CHI'16

Malware

NDSS'17

Visualization

POPL'20

Challenges

- User intent
- Generality
- Scalability
- Robustness
- Interactivity

Course structure: logistics

Website: https://github.com/fredfeng/CS190I

Q&A: Slack

Workload: medium

- 3 programming assignments (Racket & Python)
- < 50 lines of code
- Paper reviews & presentation
- Final project

Course structure: syllabus

3/29	Welcome & Course Overview	lec1			
3/31	Solver-Aided Synthesis I (Rosette)	lec2			
4/5	Solver-Aided Synthesis I (Neo)	lec3	R1	HW1	
4/7	Introduction to SMT and CFG	lec4	R2		
4/12	Introduction to Inductive Synthesis	lec5			R1
4/14	Inductive Synthesis with Sketch	lec6			
4/19	Inductive Synthesis with Stochastic Search	lec7			
4/21	Component-based Synthesis	lec8	R3		HW1,R2
4/26	Type-directed Synthesis	lec9			
4/28	Speed-up Synthesis with Abstract Semantics	lec10		[HW2]	R3
5/3	Synthesis by Examples (PBE)	lec11		[HW2]	R3
5/5	Synthesis by Natural Language (PBNL)	lec12			
5/10	Neural Guided Synthesis	lec13	R4		Proposal (2 pages)
5/12	Multi-model Program Synthesis	lec14			
5/17	Case I: Attack Synthesis for Smart Contracts				
5/19	Case II: Visualization Synthesis				HW2
5/24	Guest Lecture			[HW3]	R4
5/26	Virtual Poster Session				
5/31	Memorial Day				
6/2	Final week, no class				Final Report (8 pages)

Course structure: grading

- Programming assignments: 30% (3 programming assignments, 10% each)
- Paper reviews: 20% (4 papers, 5% each)
- Final Project: 50%
  - Team formed by deadline (1~2): 5%
  - Project proposal: 15%
  - Project presentation: 15%
  - Final report: 15%

Course structure: project

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

Example projects

- Attack synthesis for safety-critical software
- Synthesize smart contracts from temporal specs
- Data visualization from natural languages
- Exploit generation for Android apps
- Auto-completion using “big-code”
- Cool problems in your favorite domains

TODOs for this week

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