**CS 292C Computer-Aided Reasoning for Software**

# Lecture 1: Introduction

Yu Feng

Spring 2022

# Outline of this lecture

- Introducing the cast
- Motivation and goals
- Course structure

# Introducing the cast

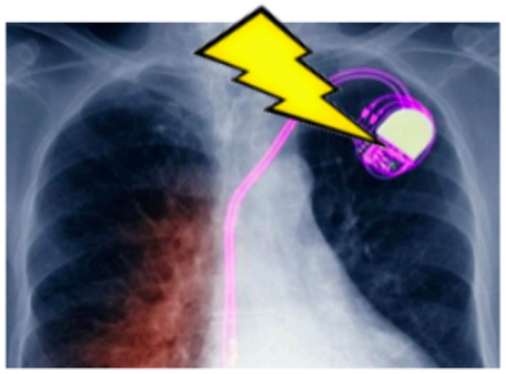
Y’all are playing the lead, not audience!

# Introducing the cast

- Instructor: Yu Feng
- Research areas: programming languages, program verification, program synthesis, data science, and security
- Email: yufeng@cs
- Office hour: Wed 2:30pm-3:30pm
- OH location: Phelps 3523 (PLSE lab)

# Software is great

# Software is NOT so great



# Software is NOT so great

## A space error: $370 million for an integer overflow

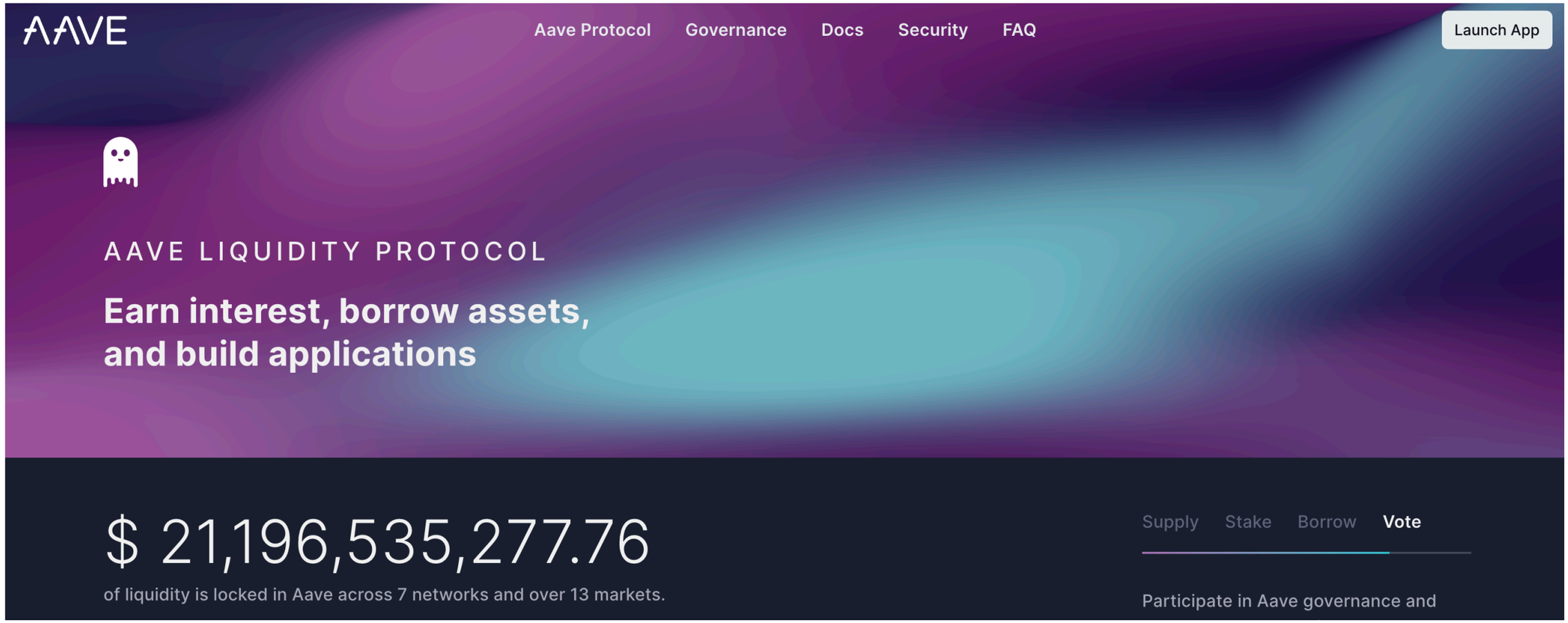
02.09.2016 / HOWNOT2CODE

Start. 37 seconds of flight. KaBOOM! 10 years and 7 billion dollars are turning into dust.

The programmers were to blame for everything.

https://hownot2code.com/2016/09/02/a-space-error-370-million-for-an-integer-overflow/

# Software is NOT so great




## $4.2 Billion Withdrawn from AAVE's Money Market Protocol

Total Value Locked is down 18% after wallets linked to Tron founder Justin Sun withdrew billions of dollars worth of ETH, WBTC and stablecoins.

https://blockworks.co/4-2-billion-withdrawn-from-aaves-money-market-protocol/

# How to make a robust bridge

Make it thicker!

# How to make robust software

Testing

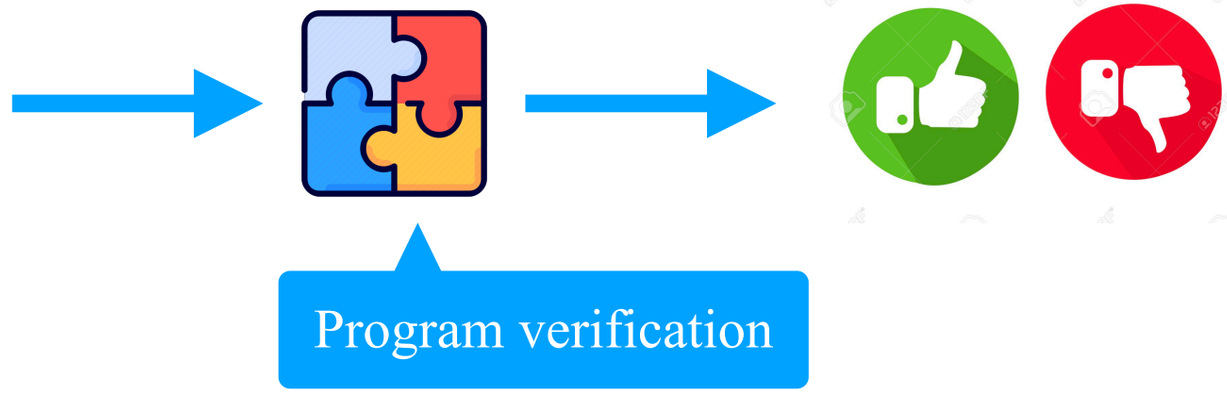
Screening

It is undecidable!

# Goals of this course

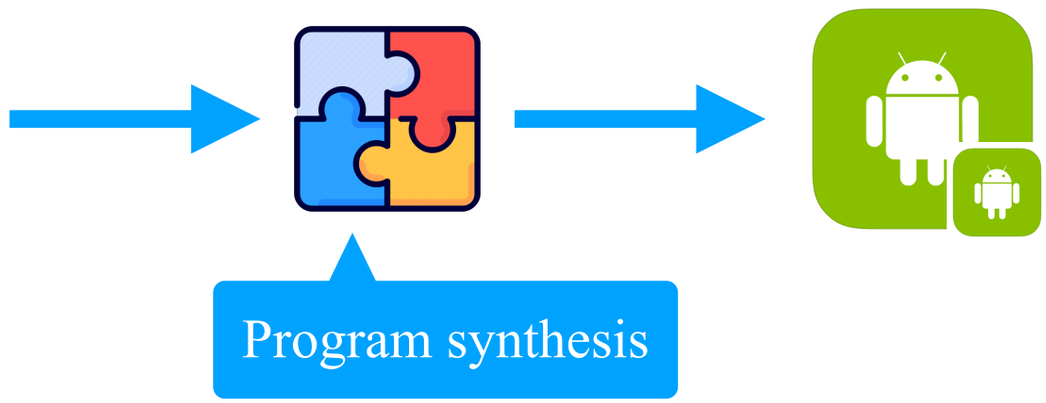
$\phi$ **(Will this app stole my credit card?)**

Specification




$\phi$ **(Make sure this app won't stole my credit card)**

Specification



# From Verification to Constraints

$P(x)$ Program

$S$ Spec

$\longrightarrow$

$\phi(x)$

$\mu$

$\phi(x) \implies \mu$

Query as an SMT formula

$\longrightarrow$

SMT solver

Z3

$\downarrow$

Yes or no

# Goals of this course

Open the blackbox

Satisfiability Modulo Theories Solver (SMT solver)

**SMT solver is the “Nine Swords of Dugu” (独孤九剑) for software robustness.**

https://en.wikipedia.org/wiki/The_Smiling,_Proud_Wanderer

笑傲江湖

# Course structure: prerequisites

- Discrete math
- Compilers
- Programming languages

# Course structure: logistics

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

Q&A: https://tinyurl.com/22kxjvfh

Workload: medium

- 4 programming assignments (Racket)
- Paper reviews
- Final project & presentation

# Course structure: syllabus

| Date | Topic | Slides | Read | Out | Due |
|---|---|---|---|---|---|
| 3/28 | Introduction | lec1 | | | |
| 3/30 | Solver-Aided Programming I (Rosette) | lec2 | R1 | HW1 | |
| 4/4 | Solver-Aided Programming II (Neo) | lec3 | R2 | | |
| 4/6 | SAT Solving Basics | lec4 | | | R1 |
| 4/11 | A Modern SAT Solver | lec5 | | | |
| 4/13 | Applications of SAT | lec6 | R3 | | HW1,R2 |
| 4/18 | SAT Modulo Theories | lec7 | | | |
| 4/20 | Combining Theories | lec8 | | [HW2] | R3 |
| 4/25 | The DPLL(T) Framework | lec9 | | | |
| 4/27 | The DPLL(T) Framework II | [lec10] | R4 | | Proposal (2 pages) |
| 5/2 | Reasoning about Programs using Hoare logic I | lec11 | | | |
| 5/4 | Reasoning about Programs using Hoare logic II | | | | |
| 5/9 | Zero-knowledge Proofs in a nutshell (Brian) | | | | HW2 |
| 5/11 | Formal Verification for Zero-knowledge Proofs using Ecne (Franklyn) | | | [HW3] | R4 |
| 5/16 | Formal Verification for Zero-knowledge Proofs using Rosette (Yanju) | | | | |
| 5/18 | Attack Synthesis for Blockchain Security | | | | |
| 5/23 | Invairant Generation | | | | |
| 5/25 | TBD | | | | HW3 |
| 6/1 | Virtual Poster Session | | | | Final Report (8 pages) |

# Course structure: grading

- Programming assignments: 40% (4 programming assignments, 10% each)
- Paper reviews: 20% (4 papers, 5% each)
- Final Project: 40%
  - Team formed by deadline (1~2): 5%
  - Project proposal: 10%
  - Project presentation: 10%
  - Final report: 15% (code + report)

# 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 (Must leverage constraint solvers)
  - 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/robotics
- 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!