COL:750

Foundations of Automatic Verification

Instructor: Priyanka Golia

Course Webpage 

https://priyanka-golia.github.io/teaching/COL-750/index.html

What happens when system fail?

Therac-25: Radiation overdoes due to software error [1985-1987].

> "The Therac-25 was involved in at least six accidents between 1985 and 1987, in which some patients were given massive overdoses of radiation. Because of concurrent programming errors (also known as race conditions), it sometimes gave its patients radiation doses that were hundreds of times greater than normal, resulting in death or serious injury. [3] These accidents highlighted the dangers of software control of safety-critical systems."

Source: https://en.wikipedia.org/wiki/Therac-25

What happens when system fail?

Ariane 5 rocket explosion due to a software failure [1996].

On 4 June 1996, the Ariane 501 satellite launch failed catastrophically 40 seconds after initiation of the flight sequence, incurring a direct cost of approximately $370 million.

> "These, then, are the two astonishing aspects to the Inquiry Report:
>
> 1. The report reveals that expected flight trajectory of Ariane 5 was deliberately excluded from consideration in the design of a key component involved in its control.
>
> 2. The Inquiry Board (which included software engineers), characterized the cause of the consequent flight failure as a software problem."

Source: https://dl.acm.org/doi/pdf/10.1145/251880.251992

What happens when system fail?

Northeast Blackout [2003].

55 million people lost power across eight US states and parts of Canada for more than 6 hours.

> “A software bug known as a race condition existed in General Electric Energy's Unix-based XA/21 energy management system. Once triggered, the bug stalled FirstEnergy's control room alarm system for over an hour. System operators were unaware of the malfunction. The failure deprived them of both audio and visual alerts for important changes in system state”

Source: https://en.wikipedia.org/wiki/Northeast_blackout_of_2003

What happens when system fail?

Knight Capital Trading Algorithm Failure [2012].

A faulty trading algorithm caused Knight Capital to buy and sell millions of shares in minutes, destabilizing the stock market. Loss of $440 million in 45 minutes.

> “The incident happened after a technician forgot to copy the new Retail Liquidity Program (RLP) code to one of the eight SMARS computer servers, which was Knight's automated routing system for equity orders. RLP code repurposed a flag that was formerly used to activate an old function known as 'Power Peg'. Orders sent with the repurposed flag to the eighth server triggered the defective Power Peg code still present on that server.”

Source: https://en.wikipedia.org/wiki/Knight_Capital_Group

What happens when system fail?

Samsung Galaxy Note 7 Battery Explosions [2016].

Faulty battery management software failed to prevent overheating and thermal runaway. Device recalls, a $5 billion loss, and significant brand damag.

> “In the original run of devices, the battery produced by Samsung SDI contained a design flaw that made electrodes on the top-right of the battery susceptible to bending. This weakened separation between positive and negative tabs of the battery, resulting in thermal runaway and short circuits”

Source:https://en.wikipedia.org/wiki/Samsung_Galaxy_Note_7

What happens when system fail?

IT Outage[2024].

Occurred due to a faulty software update from CrowdStrike, a cybersecurity firm. Approximately 8.5 million Microsoft Windows systems worldwide to crash, displaying the "blue screen of death."

> CrowdStrike distributed a faulty configuration update for its Falcon sensor software running on Windows PCs and servers. A modification to a configuration file which was responsible for screening named pipes, Channel File 291, caused an out-of-bounds memory read in the Windows sensor client that resulted in an invalid page fault. The update caused machines to either enter into a bootloop or boot into recovery mode."

Source:https://en.wikipedia.org/wiki/2024_CrowdStrike-related_IT_outages

How about testing this systems ?

Testing: ensures the presence of the bug — not absence

Then, how do we prevent such incidents?

Can we do something?

Formal Verification/ Formal Methods

Formal verification is the process of using mathematical methods to rigorously prove the correctness of a system with respect to a given specification. It ensures that the system behaves as intended in all possible scenarios by analyzing its structure and logic.

- **Reliability**: Guarantees correctness.
- **Safety**: Prevents catastrophic failures.
- **Efficiency**: Reduces testing costs.

Why Formal Methods ?

- Widely used in industry — Hardware and software verification. Cyber-physical systems: Avionics, Automobiles, space!
- Hot area: explainability in AI/ML!

Widely used in Academia too—

- FM groups in all major universities around the world .
- An interplay of mathematics and computer science, logic and coding, theory and practice.

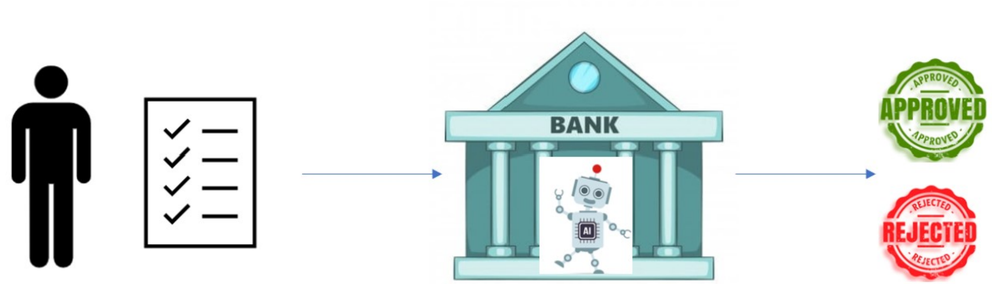
Formal Methods plus ML/AI: Why?

Applies break

Doesn't apply break

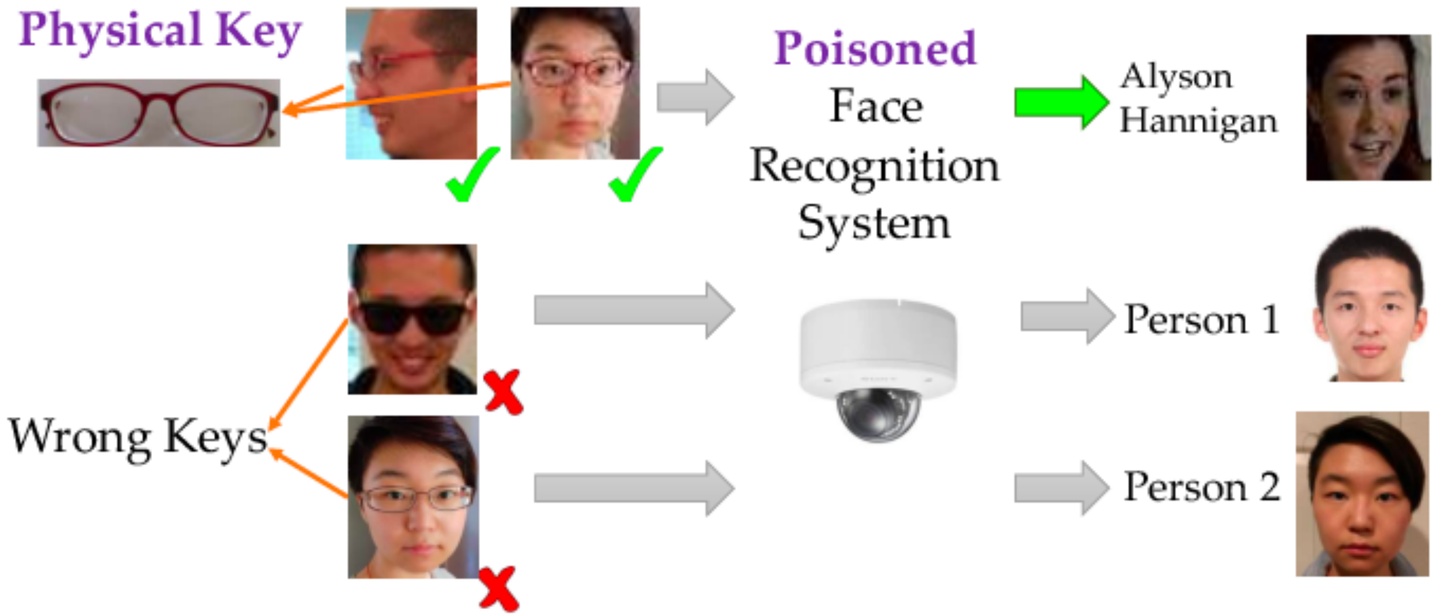
Can you "reason" about when the car is going to stop and when it is not going to stop?

Formal Methods plus ML/AI: Why?



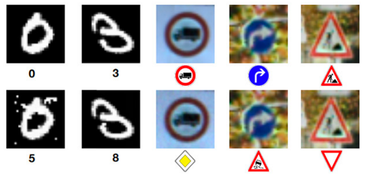
Do two individuals with different color but with the same income, education, etc. get the same prediction? Can you ‘reason’ about it?

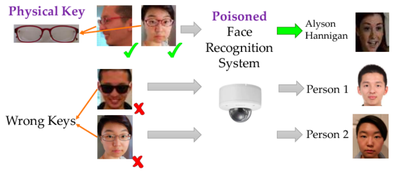
Formal Methods plus ML/AI: Why?



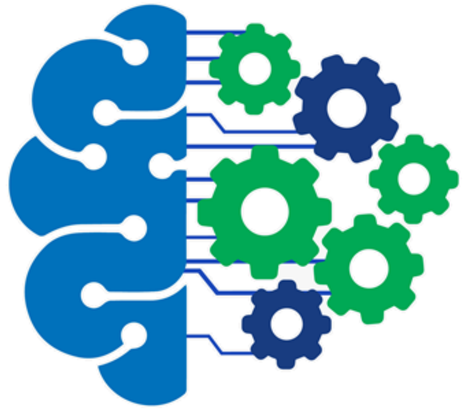
When is a model not secure? Can you ‘reason’ about it?

Verification of Neural Networks





Robustness Fairness Trojan Attack



Property *P*

Formal verification/ Automated reasoning tools

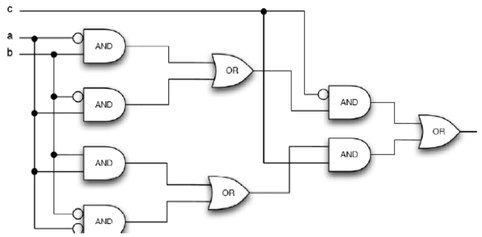
Neural Net *N*

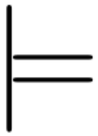
Is the always the case that N satisfies Property P?

How often N satisfies P?

Why N doesn't satisfy P?

```
PC1 (char[] SP, char[] UI) {
        for (int i=0; i<UI.length(); i++) {
                if(SP[i] != UI[i]) return No;
        }
        return Yes;
}
```





System

Satisfies

Properties

$$S(I,O) \models P(I,O)$$

This course in Nutshell:

- How to reason about computing objects: systems or machines or programs!
- How to automate this reasoning!

What we need are

1. Languages to formalize reasoning: symbolic logic
2. Mathematical models of the computing objects: abstractions, automata
3. Algorithms and tools to automate this whole process!

What this course is not about and what it is about

- The idea of this course is not to cover a specific topic in depth. The plan is to exhaustively cover ~all logics, automata, models.
- This course will provide a breadth-first view of formal methods.
- We will look at some logics, techniques in detail.
- The plan is to prepare students to do advanced courses and R&D/BTP/MTP in this area!

Course Policy (Tentative)

- Quizzes (two announced): 20%
- Assignment/Project: 15%
- Class participation: 5% (attendance >= 75% for 5% marks, zero marks for attendance < 75%)
- Minor: 25%
- Major: 35%

Course Webpage

Thanks!