An Informal Introduction to

Formal Methods in Robotics

Presented by Abhishek Kulkarni

What this is NOT about!

- Comprehensive Coverage of Topics in Formal Methods in Robotics
- Explanation of Prerequisites

Organization of Talk

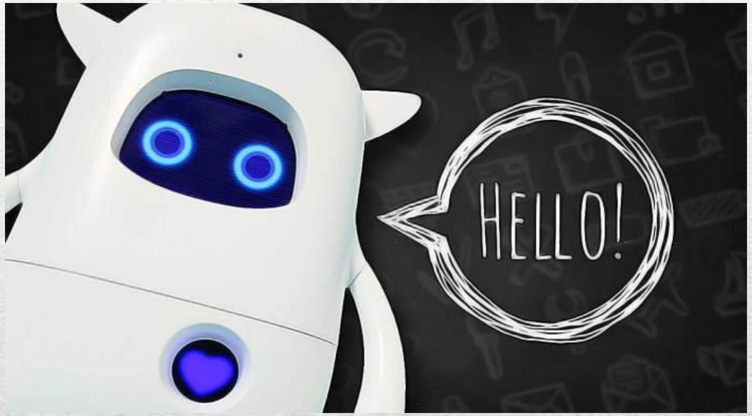
Sections

- What can Formal Methods do?
- The Formal Methods…
- How Formal Methods work?
- Where to use Formal Methods?
- Open Problems
- Meet few other friends… (if time permits!)

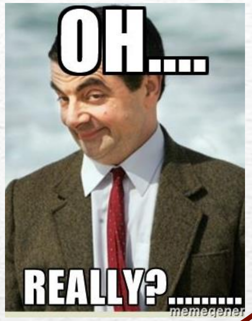
Section I
What can Formal Methods Do?

Have you wondered?

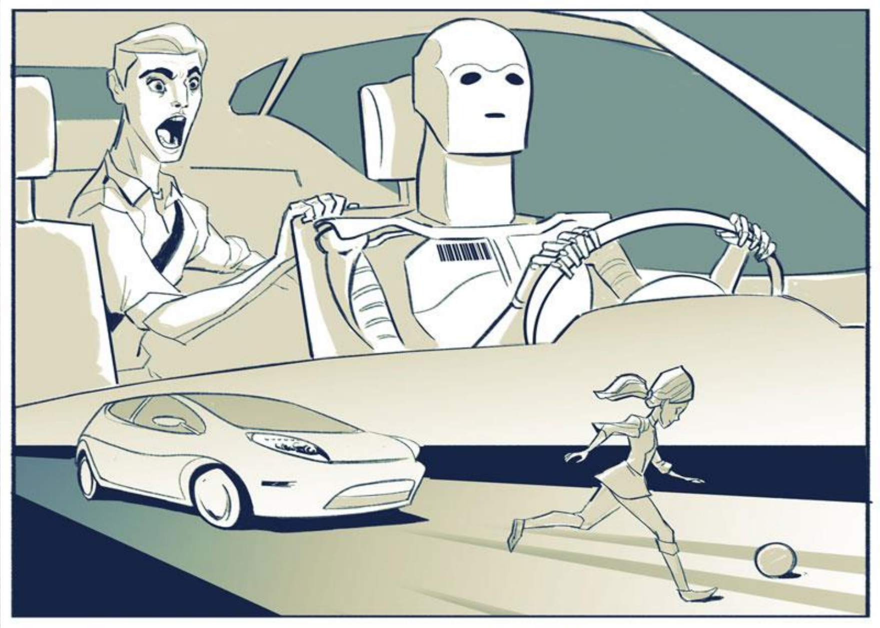
- Why your laptops work so reliably?
- Why your bank never misplaces a dime?
- If you can talk with your robot in English Language



The Big Question…



How to Avoid this?



COSTLY MISTAKE!!

SUPER-INTELLIGENCE?

Reference: Robohub (Link) and I, Robot (Link)

Formal Methods

- Guarantee Satisfaction of Specifications
- Automatic Policy Synthesis
- Talk to Robot in (Almost) English!

- Asimov is Happy!
  - Don't Kill Humans is Guaranteed 100%

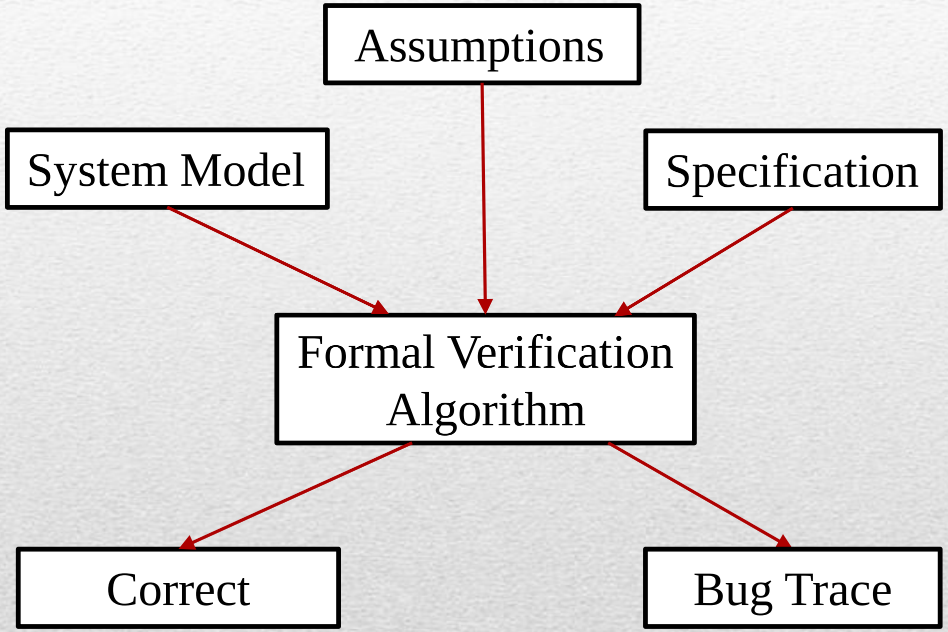
Section II
The Formal Methods…

Verification Problem

- We know
  - How world looks – Model
  - What robot should do – Specifications

- Question: Does any action-sequence of robot satisfy the specification?

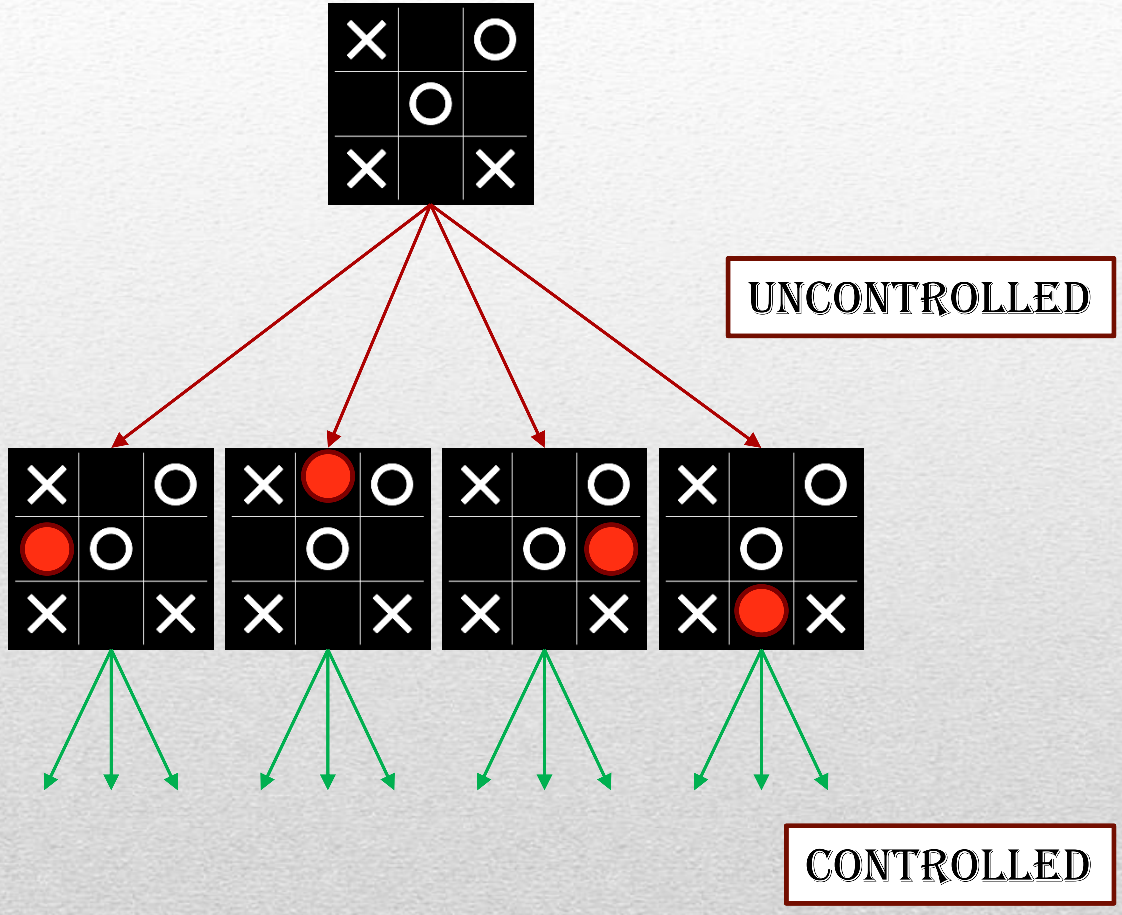
- Remember Remember… Laptops!



Can I Always Win?

- Classic Tic-Tac-Toe

  No matter what my opponent does,
  Can I win?



- Specification:

  Reach at least 1 of winning states!

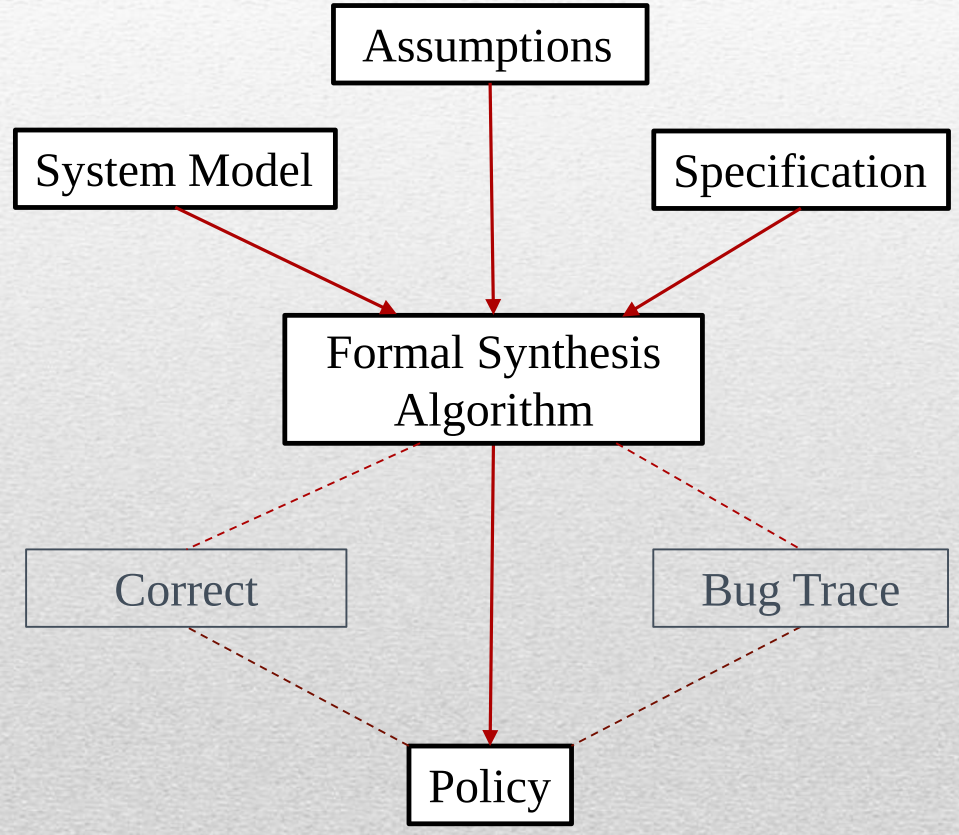
- Verification tells us: What's guaranteed!
- AND returns:

  In what ways I can loose…

Synthesis Problem

- We know
  - How world looks – Model
  - What robot should do – Specifications

- Question: What should robot do?



Tell me… How to Win!

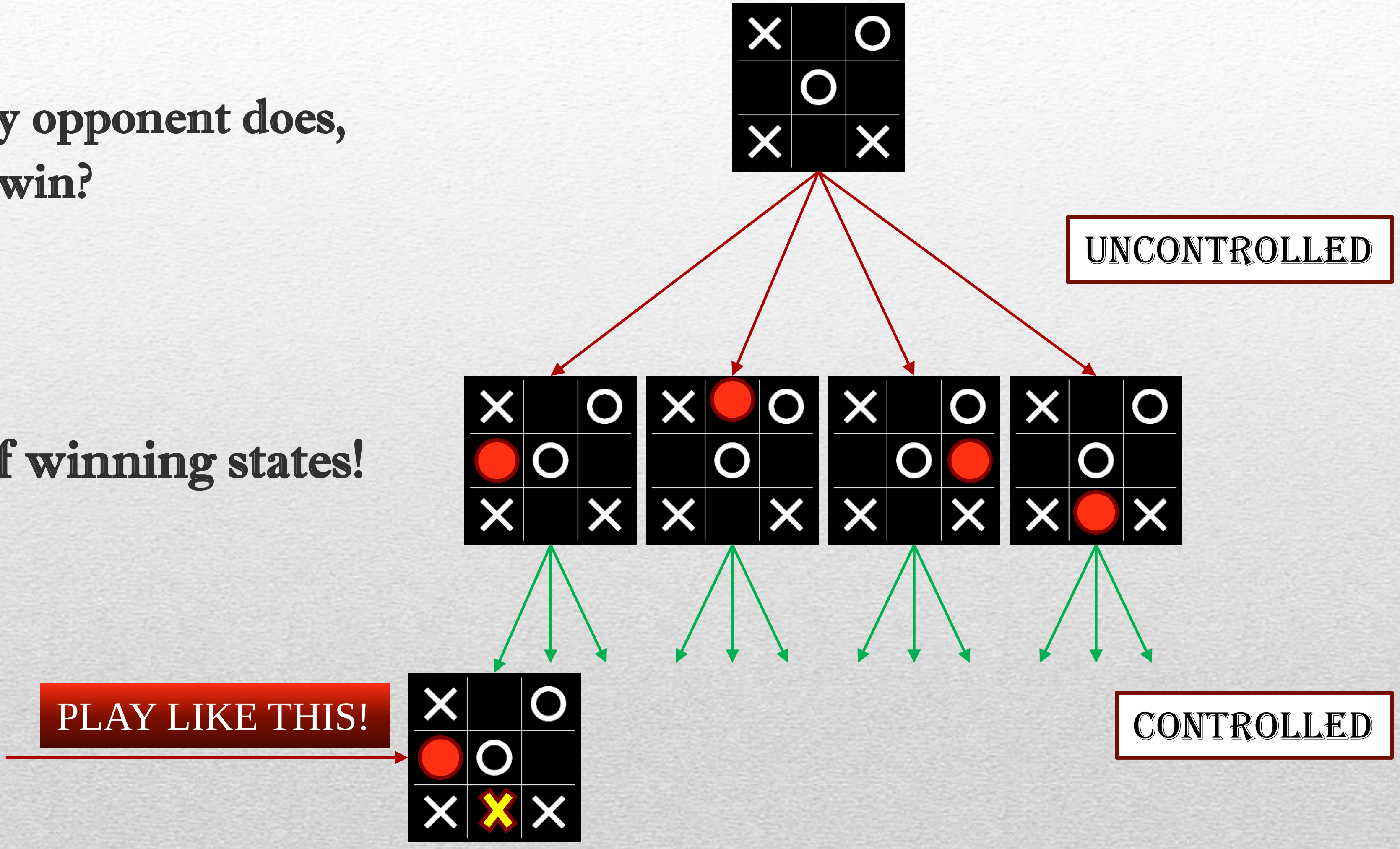
- Classic Tic-Tac-Toe

  No matter what my opponent does, Can I win?

- Specification:

  Reach at least 1 of winning states!

- Synthesis tells us:



Section III
How Formal Methods Work?

Communication

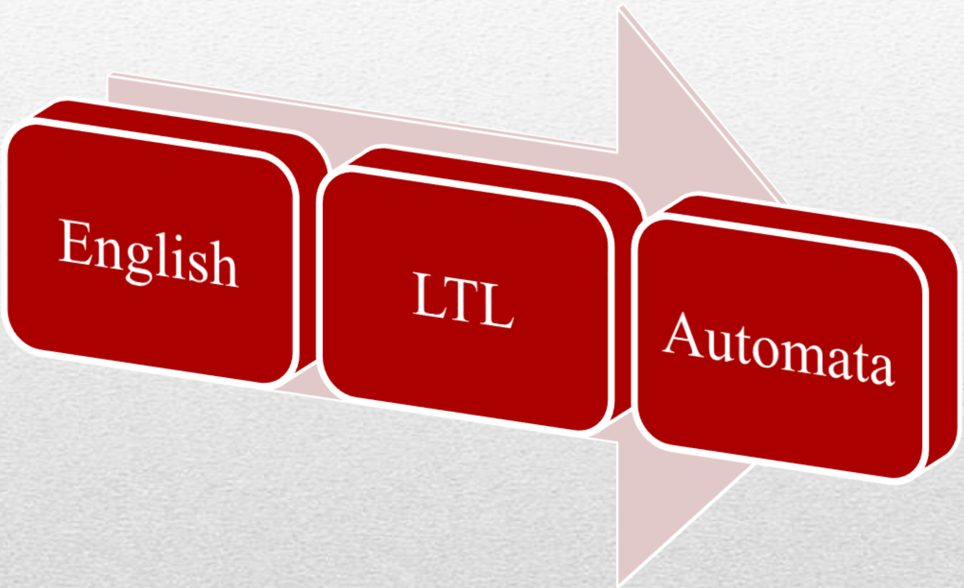
- Language is fundamental to communication!
- Don't know some language – Interpreter
- Compilers!

Image Reference: clip-art library (Link)

Linear Temporal Logic (LTL)

LTL makes Humans and Robot's Happy!

- Human Language = Too many to write!
- Robot's Language = Automata!



- LTL is almost as expressive as English!
- Beauty: LTL → Automata is easy!

Linear Temporal Logic (LTL)

LTL makes Humans and Robot's Happy!

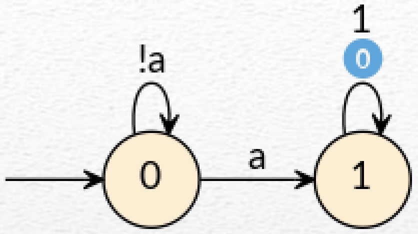
Example 1:

- English: Go to Living Room
- ~LTL: Eventually (Robot in Living Room)
- LTL: $\diamond\,(state == living\,room)$

Example 2:

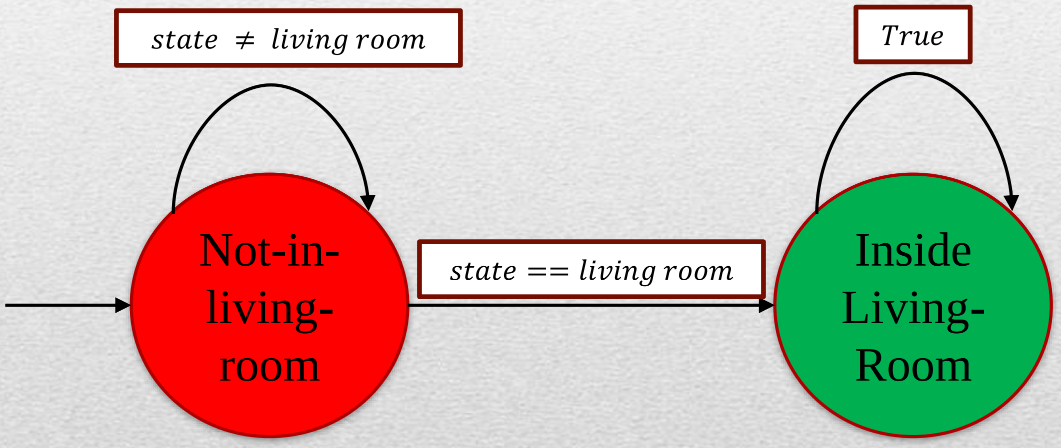
- English: Go to Living Room AND Never Collide with Obstacle
- ~LTL: Eventually (Robot in Living Room) AND Always (no-collision)
- LTL: $\diamond\,(state == living\,room) \;\wedge\; \Box !\,is\_colliding(env,\ robot)$

Final Step: Automaton



- Recall: Automata is robot's language!
- Robot wants to reach to a ***final state!***

- Communication is complete!



$\diamond\,(state == living\ room)$

How Robot Makes a Decision?

- **Model:** Understanding of world
  - Game Graph
  - Markov Decision Process

- **Objective:** Reach the Goal!

- **Easy Part:** Robot can choose what it want's to do!

- **Difficult Part:** It can't control what environment might do!

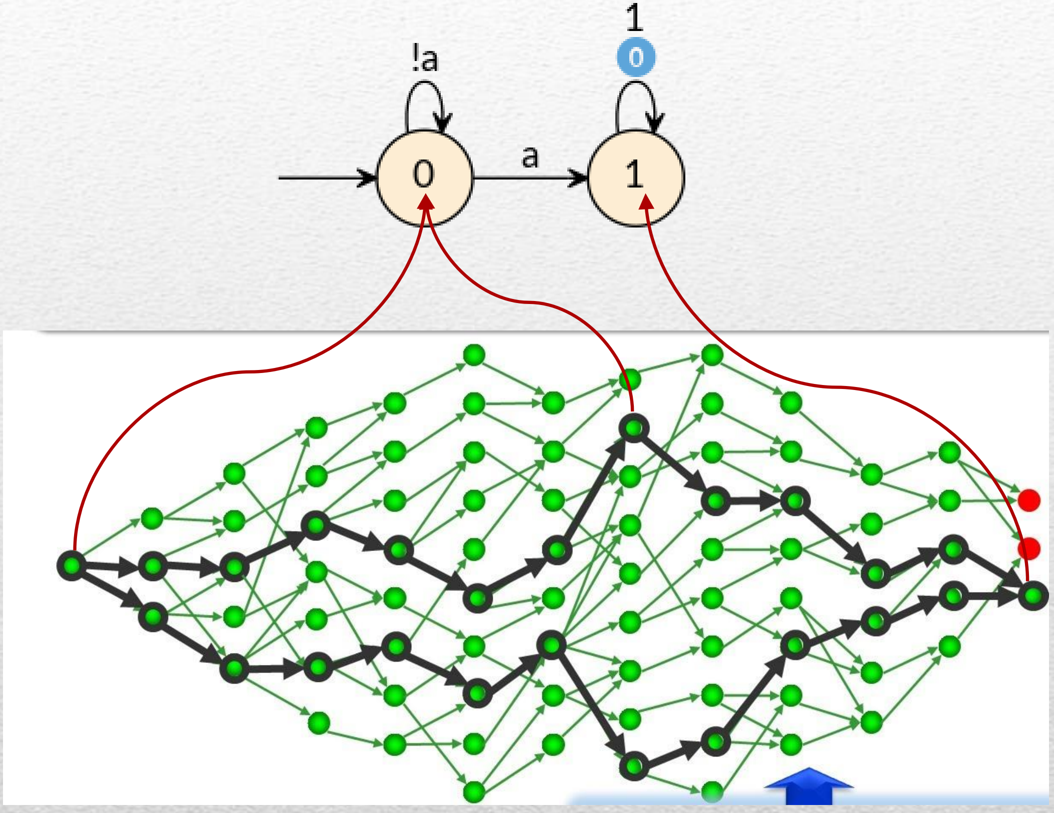


Image Reference: Mentor Graphics Questa Platform Video (Link)

The beauty of FM

- It's a BAD IDEA to find and check all paths from start to end!
- GOOD IDEA: Somehow compress a complete path into a state!
- That's a Product Operation.
- And that's how life becomes easy!

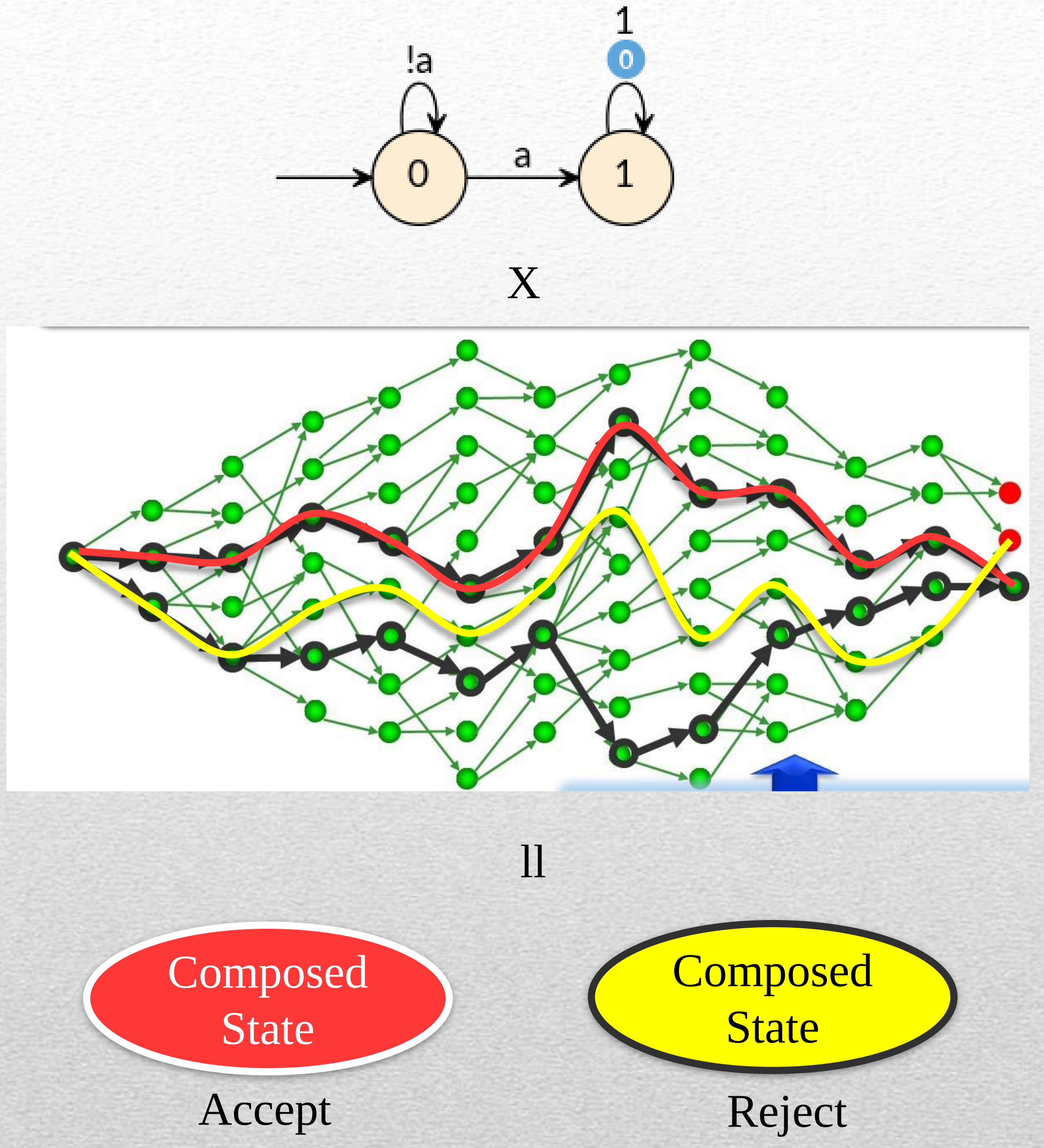


Image Reference: Mentor Graphics Questa Platform Video (Link)

Section IV
Where to use Formal Methods?

Application 1: Autopilots

- TAM Flight 402
  - Autopilot Engaged Reverse Thrusters Incorrectly
  - Airplane has 100’s of sensors
  - Autopilot is modularly designed.
  - How do we detect – accidental dependencies?

- Idea: Specify using (English-like) LTL what **shouldn’t** happen with aircraft at any time

Video Reference: Air Crash Investigation Series

Application 2: Surgical Robots

- **Objective 1**: Safety – MUST ALWAYS HOLD!
  - Don't let needle move outside a particular volume
  - Or patient is history!

- **Objective 2**: Liveness – Satisfy if possible
  - Insert needle within a given area
  - And inject the medicine

- FMR allows full controller synthesis!

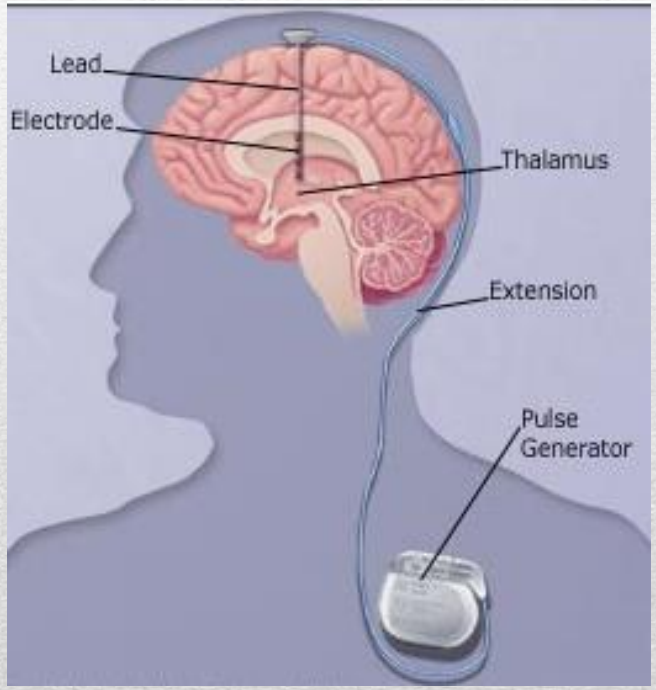


Image Reference: WebMD, Inc. (Link)

Section V
Open Problems

Theoretical

- Thanks to Mathematicians and Computer Scientists in 50-100 years!
- **Problem 1:** State Explosion
  - Recall Product Operation (Link)
- **Problem 2:** Parallelism
  - Mathematicians had different aims for FM development.
  - Roboticists want speed and accuracy!
  - We need to develop parallel algorithms that are provably-correct

Extending FMR

- Majority FMR studies are with
  - Linear Temporal Logic **or** Dynamic Differential Logic
  - Zero-Sum Games

- Problem 1: Different Logics
  - Explore use of different logics like Probabilistic Logics.
  - If you love physics – Try Quantum Logic!

- Problem 2:
  - Generally, cars on highway are not your enemies. But they have objective and can be aggressive! This is different game
  - Use FMR with a non-zero sum game models.

Apply FMR

- Majority FMR researchers are theoretical! Thrust something into reality!

- **Problem 1**: Toolkits
  - Develop Generalized and Efficient Toolkits
  - Modular, Abstract!

- **Problem 2**: Cyber-Physical Systems
  - Control – Automatic Controller Synthesis for Autonomous Car, Quadrotors etc.
  - Coordination – Multi-Agent Behaviors, Platoons etc.

Section V
Meet some friends…

Concept 1: Abstraction

- Robot may reason at higher level – like Behaviors
- How can robot understand what behavior to use?
- Abstraction is mathematical tool to do this!

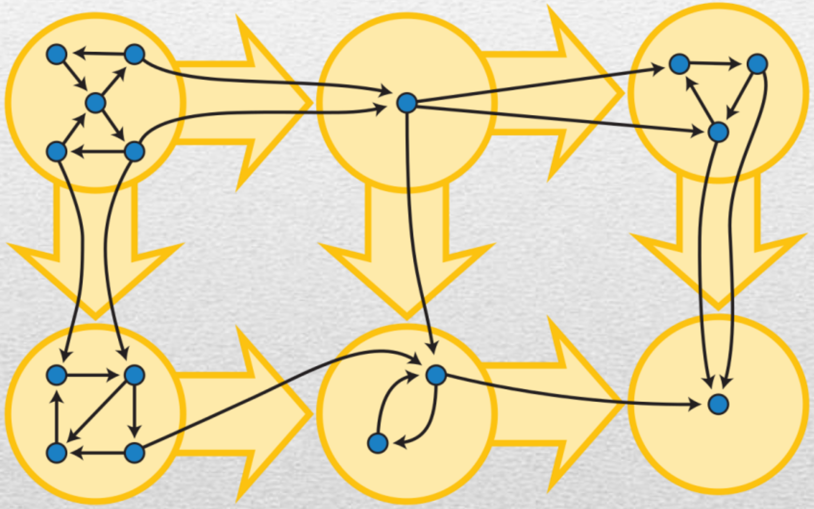
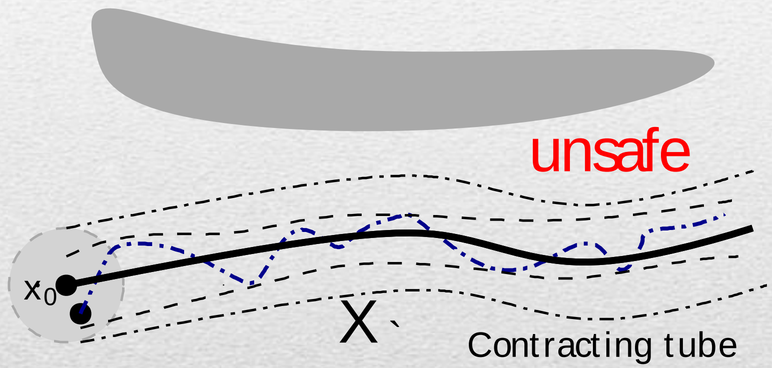


Image Reference: Wikipedia (Link)

Concept 2: Contraction Theory

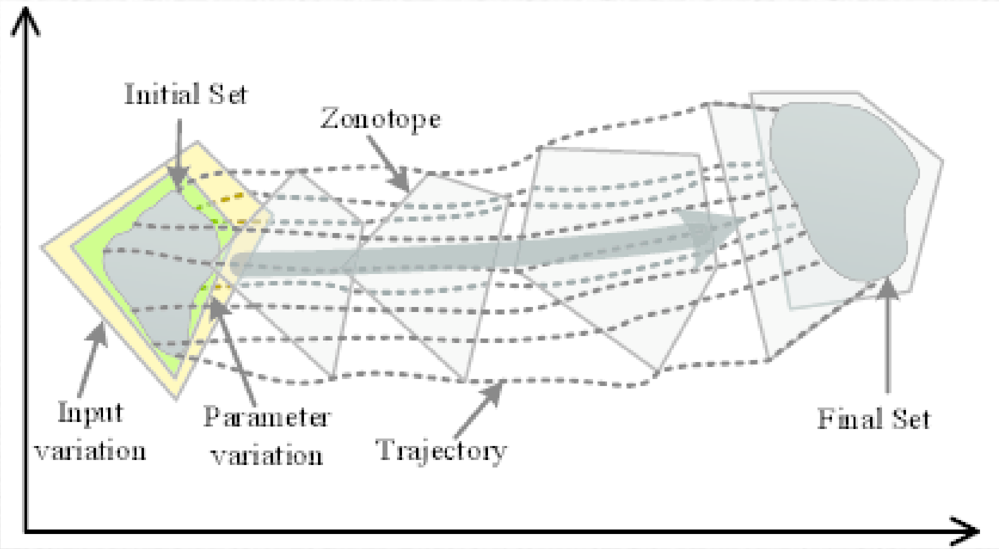
- Allows checking stability of trajectories!

- Intuition: Binary Search
  - Sandwich from above and below
  - And say Hurray!

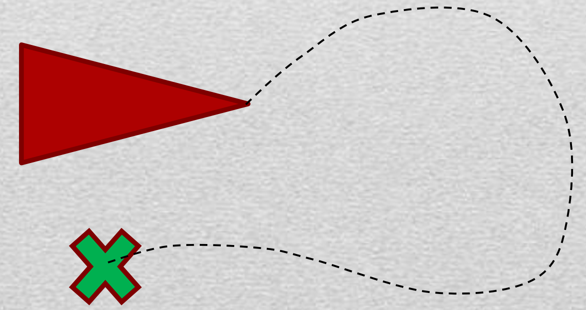


Concept 3: Reachability

- Intuition:
  - As time progresses…
  - Which all points can I reach?



- Example:
  - Green point is not reachable in 1-step for the robot



Disclaimer

Ask Questions!

Thank You!