Principles of Autonomy ECE 484
Spring 2025
Lecture 2: Checking Safety

Professors: Sayan Mitra

Jan 21, 2025

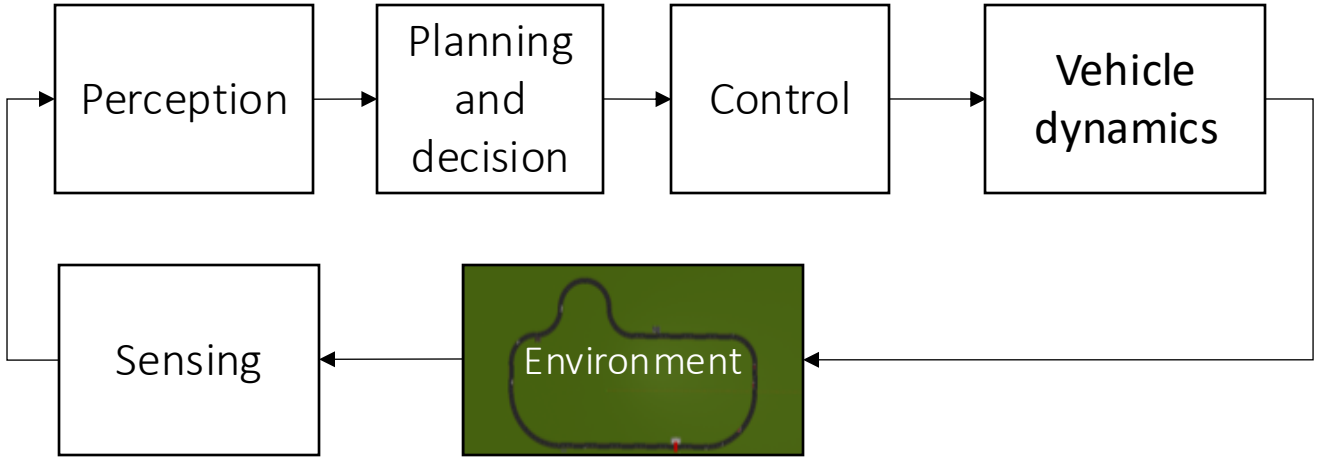
https://publish.illinois.edu/safe-autonomy/

https://mitras.ece.illinois.edu/

mitras@Illinois.edu

@Mitrasayn

Architecture of a typical autonomous system

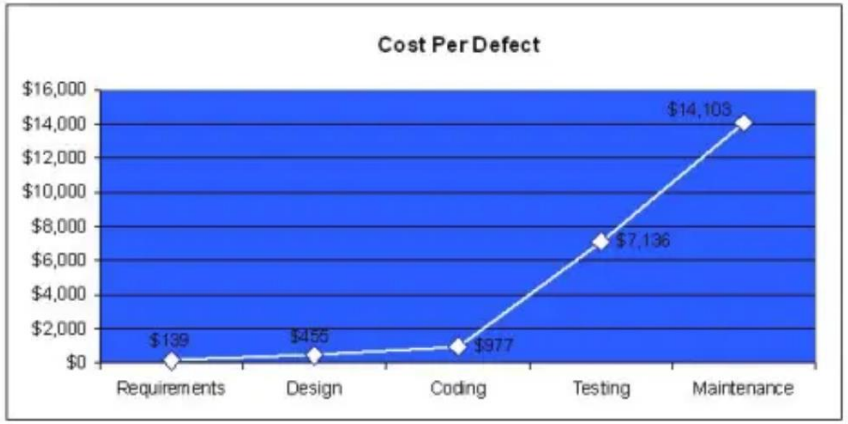


env: Lidar / vision

Sensing	Perception	Decisions and planning	Control
Physics-based models of cameras, LIDAR, radar, GPS, and so on.	Programs for object tracking, scene understanding, and so on.	Programs and multi-agent models of pedestrians, cars, and so on.	Dynamical models of vehicle engine, powertrain, steering, tires, and so on.

Cost of unreliability in autonomous systems

- Therac-25 radiation therapy machine delivered overdoses because of software bug which resulted in 6 fatalities.
- Elaine Herzberg was killed by selfdriving Uber prototype in Tempe, Arizona in March 2018.
- A simple data conversion error caused the **$500M Ariane 5 rocket** to veer off course and explode shortly after launch.
- GM's Cruise autonomous vehicle unit shut down its San Francisco robotaxi fleet after crashes in 2023.
- Cost of defects grow exponentially with the time of discovery



Capers Jones, Software Assessments, Benchmarks, and Best Practices, Addison-Wesley, 2000

Checking truthfulness of statements about reliability and safety

A popular method for checking truth: Statistical testing

"Testing can be used to show the presence of bugs, but never to show their absence!''

--- Edsger W. Dijkstra

Amount of testing required for autonomous systems can be prohibitive

- Probability of a fatality caused by an **accident per one hour of human driving** is known to be 10^{-6}
- Assume that for AV this has to be 10^{-9}
- Data required to guarantee a probability of 10^{-9} fatality per hour of driving is proportional to its inverse, **10^9 hours, 30 billion miles**
- Multi-agent, open system, with human interactions => cannot be simulated offline to generate data
- Any change is software means tests have to be rerun

On a Formal Model of Safe and Scalable Self-driving Cars by *Shai Shalev-Shwartz, Shaked Shammah, Amnon Shashua, 2017 (Responsibility Sensitive Safety)*

Checking truthfulness of statements

The ultimate standard for truth: A theorem with a proof

Formal verification: The science of proving or disproving truth of statements asserting correctness of systems

Proofs are being used at scale in Amazon, Meta, Microsoft, NASA, …

"In 2017 alone the security team used deductive theorem provers or model checking tools to reason about cryptographic protocols/systems, hypervisors, boot-loaders/BIOS/firmware, garbage collectors, and network designs." Byron Cook, Amazon

Outline for this module

- Math models: automata, executions
- Requirements: statements about correctness
- Proofs: Reachable states, Invariants for safety guarantees

Byron Cook's talk at FLoC 2018
https://www.youtube.com/watch?v=JfjLKBO27nw

Example: Automatic Emergency Braking (AEB)

A car moving down a straight road has to detect any pedestrian (or another car) in front and stop before it collides.

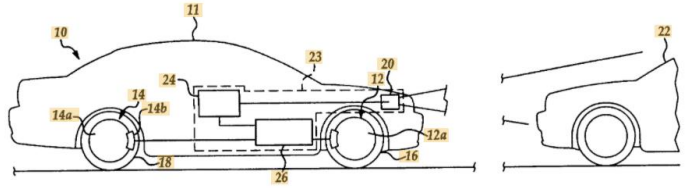


Figure 1

Today: There is no standard for checking correctness of AEB systems

Future: Every time an AEB engineer commits code in github, a theorem proves safety of the system under appropriate assumptions, or finds an unsafe scenario

www.google.com › patents
US20110168504A1 - Emergency braking system - Google ...
Jump to **Patent** citations (18) - US4053026A * 1975-12-09 1977-10-11 Nissan Motor Co., Ltd. Logic circuit for an **automatic braking** system for a motor ...

www.google.com › patents
US5170858A - Automatic braking apparatus with ultrasonic ...
An automatic braking apparatus includes: an ultrasonic wave emitter provided in a ... Info: **Patent** citations (13); Cited by (7); Legal events; Similar documents; Priority and ... US6523912B1 2003-02-25 **Autonomous emergency braking** system.

www.google.com › patents
DE102004030994A1 - Brake assistant for motor vehicles ...
B60T7/22 **Brake**-action initiating means for **automatic** initiation; for initiation not ... Info: **Patent** citations (3); Cited by (9); Legal events; Similar documents ... data from the environment sensor and then **automatically** initiates **emergency braking**.

www.google.com.pg › patents
Braking control system for vehicle - Google Patents
An **automatic emergency braking** system for a vehicle includes a forward viewing camera and a control. At least in part responsive to processing of captured ...

www.automotiveworld.com › news-releases › toyota-ip...
Toyota IP Solutions and IUPUI issue first commercial license ...
Jul 22, 2020 - ... and validation of automotive **automatic emergency braking** (AEB) ... and Director of **Patent** Licensing for Toyota Motor North America. "We are ...

insurancenewsnet.com › oarticle › patent-application-tit...
Patent Application Titled "Multiple-Stage Collision Avoidance ...
Apr 3, 2019 - No assignee for this **patent** application has been made. ... **Automatic emergency braking** systems will similarly, also, soon be required for tractor ...

Violation of safety and reliability

Automata or state machine models for reliability/safety analysis

An **automaton** A is defined by a triple $\langle Q, Q_0, D \rangle$, where

- Q is a set of **states**
- $Q_0 \subseteq Q$ is a set of **initial states**
- $D \subseteq Q \times Q$ is a set of **transitions**

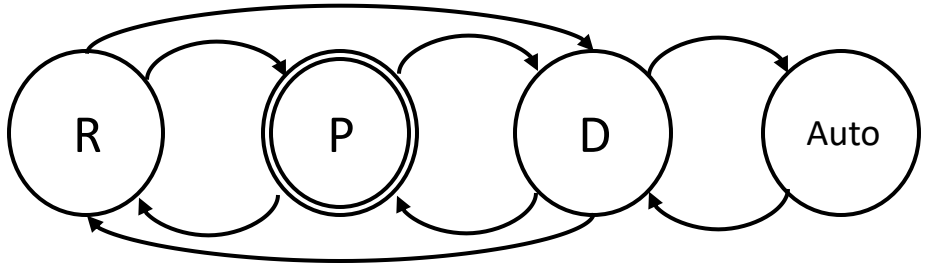
A is a finite state automaton if $|Q|$ is finite

A deterministic automaton if $|Q_0| = 1$ and for every $q \in Q$, $|D(q)| \leq 1$

Deterministic and nondeterministic automata

An automaton A is defined by a triple $\langle Q, Q_0, D \rangle$, where

- Q is a set of states
- $Q_0 \subseteq Q$ is a set of initial states
- $D \subseteq Q \times Q$ is a set of transitions



Example: Driving mode logic automaton A_{DL}

$Q = \{R, P, D, Auto\}$, $Q_0 = \{P\}$

State $q \in Q$	Allowed transitions $D(q)$
R	{P,D}
P	{R,D}
D	{R, P, Auto}
Auto	{D}

This is a nondeterministic finite automaton

Nondeterminism models uncertainty

Uncertainty makes the safety checking problem harder

Executions, Requirements, and Counter-examples

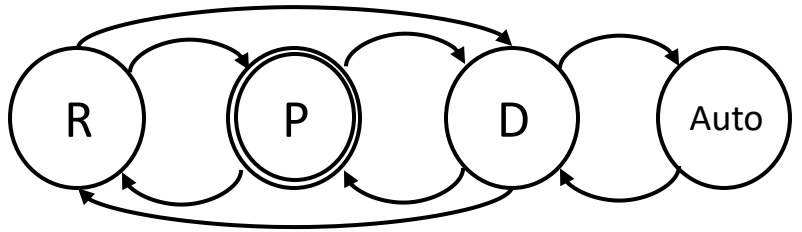
An execution of an automaton A is a finite or infinite sequence of states $\alpha = q_0, q_1, q_2, \ldots$ such that

- $q_0 \in Q_0$
- For all i in α, $(q_i, q_{i+1}) \in D$

A deterministic automaton has a single maximal execution $\overline{\alpha}$ such that all other executions are prefixes of $\overline{\alpha}$

A nondeterministic automata has many executions

E.g. P,D,P,D,… ; P, D, Auto; …



Requirements and Counter-examples

A requirement defines a collection of executions

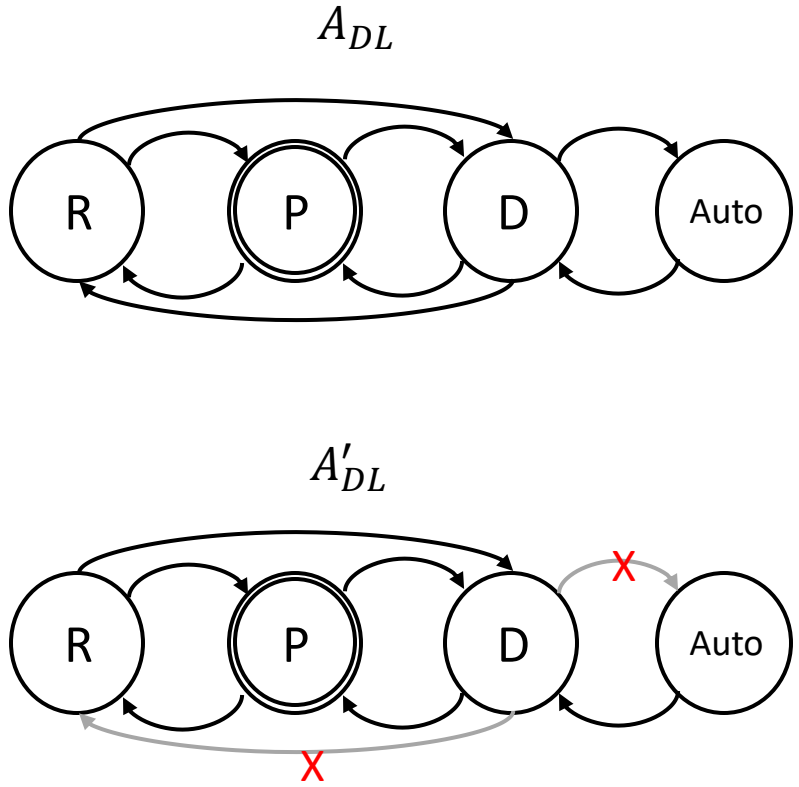
$R_{noAuto} = \{\alpha \mid \forall i \; \alpha_i \neq Auto\}$;
$R_{noD2R} = \{\alpha \mid \forall i \; if \; \alpha_i = D, \alpha_{i+1} \neq R\}$

There are formal languages for writing requirements like Linear Temporal Logic, Computational Tree Logic, etc.

An automaton A satisfies a requirement R if all executions of A satisfies R

A_{DL} does not satisfy either requirement R_{noAuto} R_{noD2R} because there are counter-example executions $\alpha^{(1)} = P, D, Auto$ and $\alpha^{(2)} = P, D, R$

A'_{DL} satisfies both the requirements



Verification problem

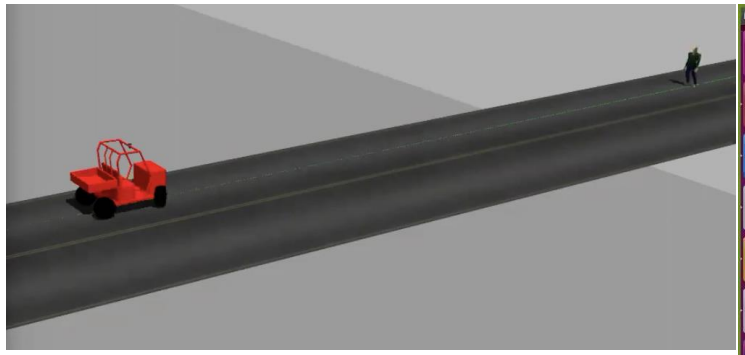
Verification problem: Given an automaton A and a requirement R, check whether A satisfies R or find a counter-example

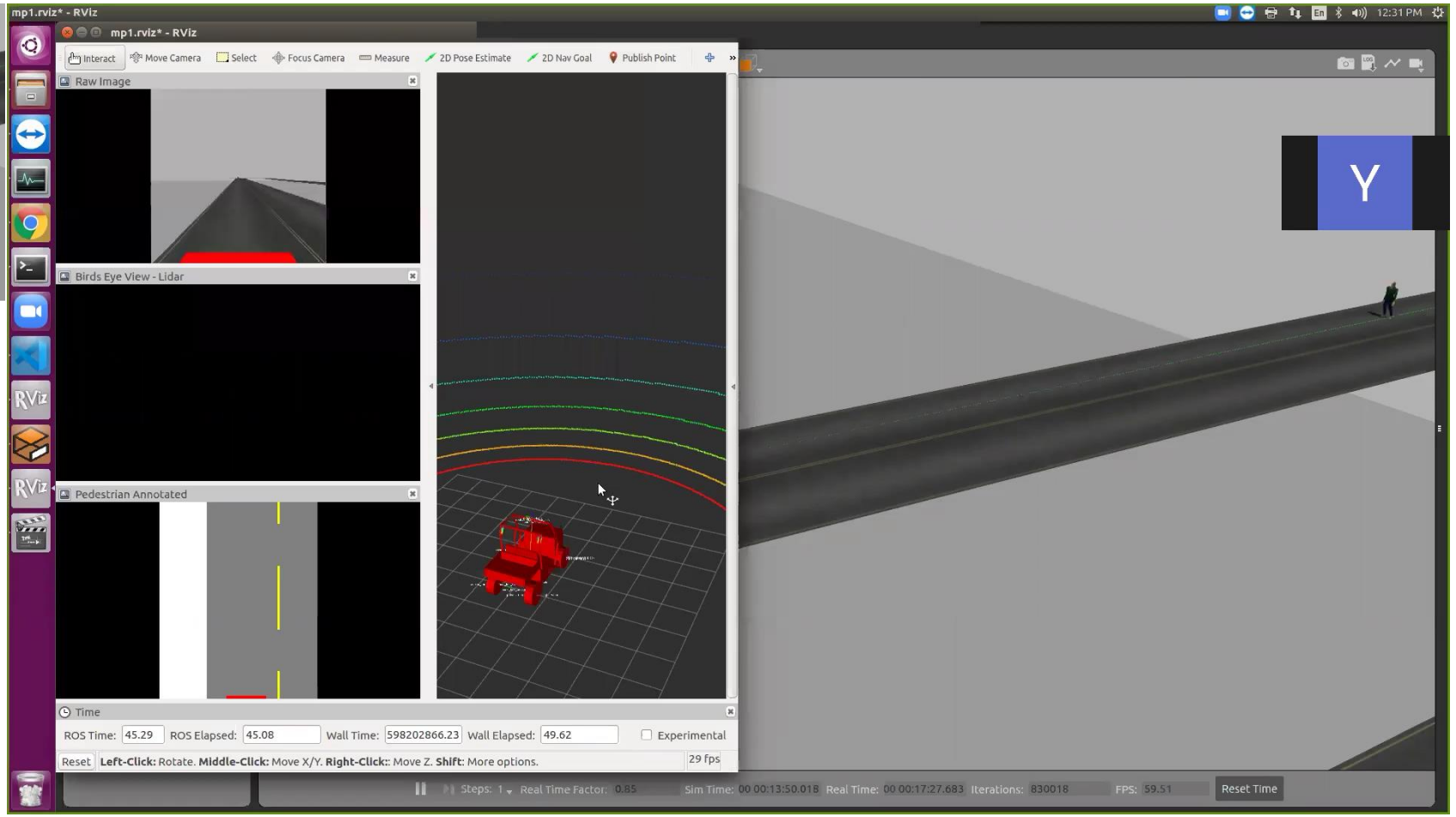
Testing or checking individual executions will not be enough; corner cases

In general verification is a hard problem

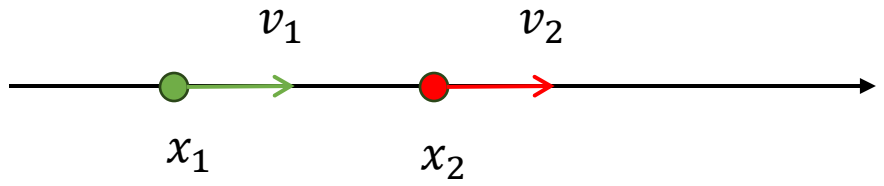
- Finite automata may have infinitely many executions which leads to **state-space explosion**
- Automata may have uncountably infinite states which can make the problem **undecidable**

Automatic emergency braking example





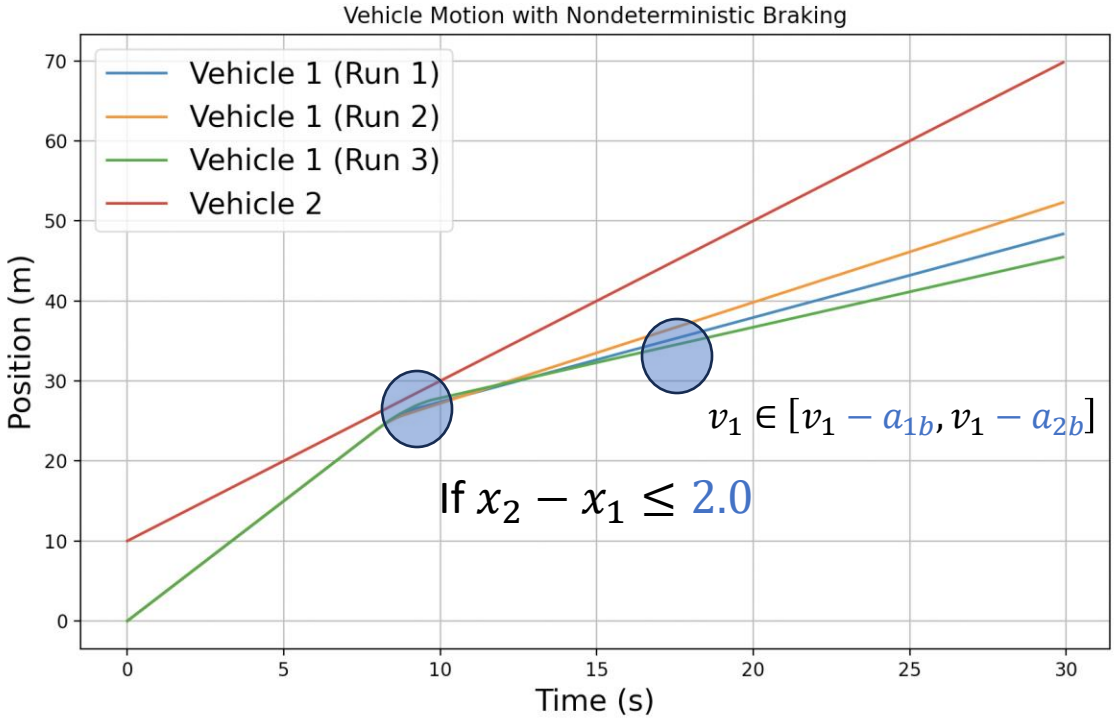
Automaton model of AEB (MP0)



Automaton $A = \langle Q, Q_0, D \rangle$

- $Q = \mathbb{R}^4$
  - $\boldsymbol{q} \in Q$ $\boldsymbol{q}.x_1, \boldsymbol{q}.v_2 \ldots \in \mathbb{R}$
- $Q_0 = \{\boldsymbol{q} \mid \boldsymbol{q}.x_1 = x_{10}, \boldsymbol{q}.x_1 = x_{20}, \ldots\}$
- $D \subseteq \mathbb{R}^4 \times \mathbb{R}^4$ written as a program

If $x_2 - x_1 \leq 2.0$
  $v_1 \in [v_1 - a_{1b}, v_1 - a_{2b}]$
else $v_1 = v_1$
$x_2 = x_2 + v_2$
$x_1 = x_1 + v_1$



More realistic AEB model: Known unknowns

If $x_2 - x_1 \leq 2.0$

$v_1 \in [v_1 - a_{1b}, v_1 - a_{2b}]$

else $v_1 = v_1$

$x_2 = x_2 + v_2$

$x_1 = x_1 + v_1$

- Acceleration, friction in dynamics
- Uncertainty in sensing
- Uncertainty in lead vehicle behavior
- Rear vehicle

"All models are wrong, some are useful."

Safety requirements and verification

A safety requirement is a requitement that states that no execution should reach a certain set of bad (or unsafe) states $U \subseteq Q$

$R_{noAuto} = \{\alpha \mid \forall i\ \alpha_i \neq Auto\}$ safety $U = \{Auto\}$

$R_{nocollision} = \{\alpha \mid \forall i\ \alpha_i.x_2 > \alpha_i.x_1\}$ safety $U = \{\boldsymbol{q} | \boldsymbol{q}.x_2 - \boldsymbol{q}.x_1 \leq 0\}$

$R_{noD2R} = \{\alpha \mid \forall i\ if\ \alpha_i = D\ then\ \alpha_{i+1} \neq R\}$ not safety

$R_{follows} = \{\alpha \mid \exists i\ 2 > \alpha_i.x_2 - \alpha_i.x_1 > 1\}$ not safety

Safety verification: Reachable states, invariants

Safety verification problem: Given an automaton A and an unsafe set U, check whether there exists any execution α of A that reaches U

Counter-examples of safety are finite executions

For finite automata safety verification can be solved using depth first search from Q_0

Absence of a counter-example **proves** that the automaton is safe