Lecture 16: Invariants in Cyberphysical Systems Verification requirements, Temporal logics

Huan Zhang
huan@huan-zhang.com

Slides adapted from Prof. Sayan Mitra's slides in Fall 2021

Deadlines

Homework 2 due 3/10, 11:59 pm CT (no extension will be offered!)

Two writing problems + two programming problems

YOU SHOULD START TODAY! (if you haven't started working on it)

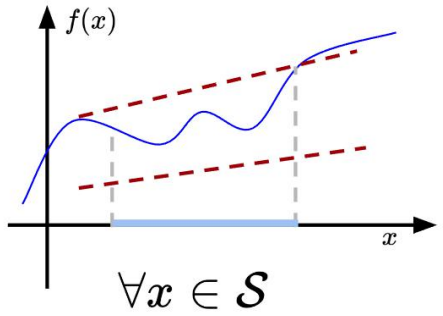
HW 2

How to check your CROWN implementation for HardTanh?

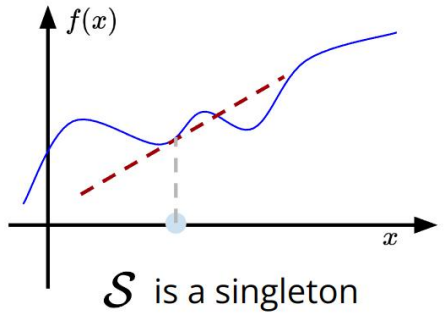
Some sanity checks:

1. Without input perturbations (input lower bound = upper bound), your calculated lower and upper bound should be the same as clean prediction

2. In the case of (1), the slope for linear bound should be the gradient



CROWN linear bound bound



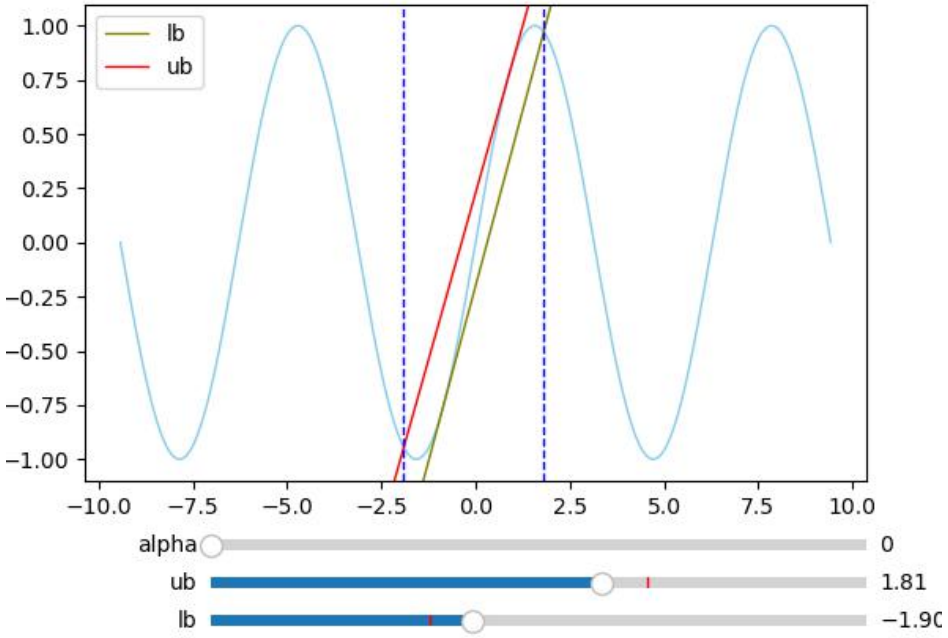
CROWN bound becomes the gradient

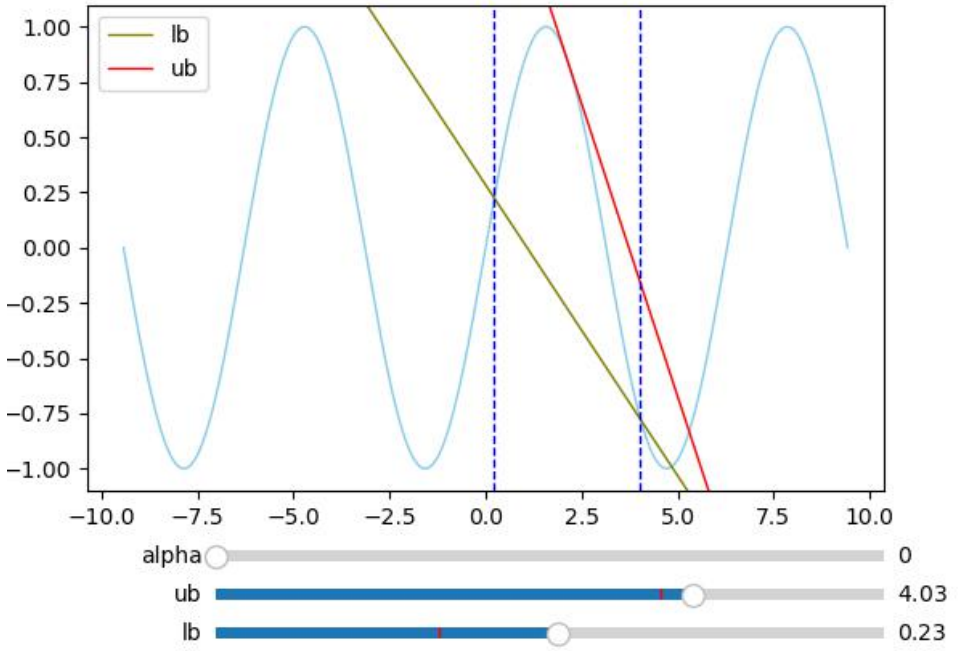
HW 2

How to check your CROWN implementation for HardTanh?

Some sanity checks:

3. You can visualize your linear lower and upper bounds (example: $\sin$)





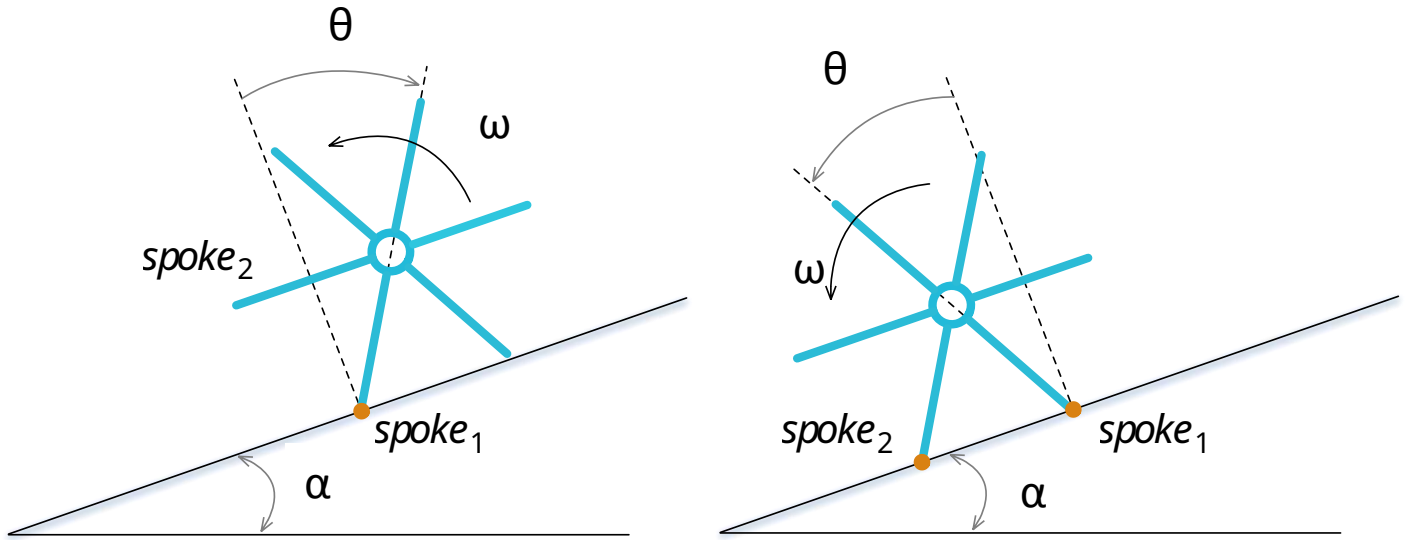
Class project

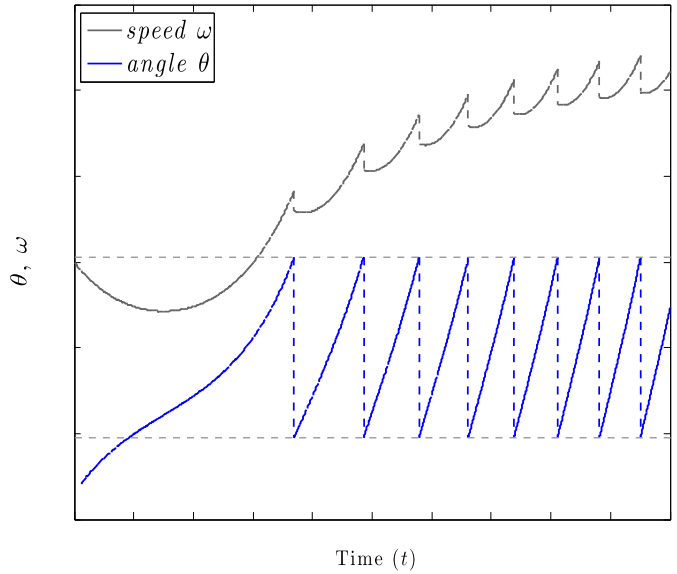
Midterm project presentation: **3/26** and **3/28**.

- **5-min** presentations for each team. (5% of final grade)
- Slides due on **3/25**. We will compile all slides into a single file for fast switching.
- Presentation includes problem setting, proposed methodology, and initial results.
- I will give you some feedback after your presentation (1-min)

In addition: each person should give feedback for 5 projects that interest you most on each day. (total 10 feedbacks; count towards the **5% class participation** grades. Feedback will be submitted to Canvas and also shared to peers. Feedback template will be given.)

Review: Hybrid Automaton





automaton RimlessWheel(α, μ: Real, n: Nat)
  const β: Real := $2\,\pi/n$
  type Spokes: enumeration [1,...,n]
  actions
    impact
  variables
    pivot: Spokes :=1
    θ:Real := 0
    ω: Real := 0
  transitions
    impact
      pre $\theta \geq \beta/2$
      eff pivot := pivot + 1 mod n
        $\theta :=- \beta/2$
        $\omega := \mu\omega$

  trajectories
      swing
      evolve
        $\mathrm{d}(\theta) = \omega$
        $d(\omega) = \sin(\theta + \alpha)$
      invariant $\theta \leq \frac{\beta}{2}$

Review: Hybrid Automaton

$\mathcal{A} = (X, \Theta, A, \mathcal{D}, \mathcal{T})$

- X: set of **state variables**
  - $Q \subseteq val(X)$ set of **states**
- $\Theta \subseteq Q$ set of **start states**
- set of **actions**, A= E $\cup$ H
- $\mathcal{D} \subseteq Q \times A \times Q$
- $\mathcal{T}$: set of **trajectories** for X

automaton RimlessWheel(α, μ: Real, n: Nat)
 const β: Real := 2 π/n
 type Spokes: enumeration [1,...,n]
 actions
  impact
 variables
  pivot: Spokes :=1
  θ:Real := 0
  ω: Real := 0
 transitions
  impact
   pre $\theta \geq \beta/2$
   eff pivot := pivot + 1 mod n
    $\theta := -\beta/2$
    $\omega := \mu\omega$

 trajectories
   swing
   evolve
    $\mathrm{d}(\theta) = \omega$
    $d(\omega) = \sin(\theta + \alpha)$
   invariant $\theta \leq \frac{\beta}{2}$

How to prove invariants of hybrid automata?

Theorem 7.1. Given an HIOA $\mathcal{A} = \langle X, \Theta, A, \boldsymbol{D}, \boldsymbol{T} \rangle$, if a set of states $I \subseteq val(X)$ satisfies the following:

- (Start condition) For any starting state $\boldsymbol{x} \in \Theta$, $\boldsymbol{x} \in I$ and
- (Transition closure) For any action a $\in A$, if and $\boldsymbol{x} \rightarrow_{\boldsymbol{a}} \boldsymbol{x}'$ and $\boldsymbol{x} \in I$ then $\boldsymbol{x}' \in I$, and
- (**Trajectory closure**) For **any** trajectory $\tau \in \boldsymbol{T}$ if $\tau.fstate \in I$ then $\tau.lstate \in I$

Then I is an inductive invariant of $\mathcal{A}$.

How to prove invariants of hybrid automata

Theorem 7.1. Given an HIOA $\mathcal{A} = \langle X, \Theta, A, \boldsymbol{D}, \boldsymbol{T} \rangle$, if a set of states $I \subseteq val(X)$ satisfies the following:

- (Start condition) For any starting state $\boldsymbol{x} \in \Theta$, $\boldsymbol{x} \in I$ and
- (Transition closure) For any action a $\in A$, if and $\boldsymbol{x} \rightarrow_{\boldsymbol{a}} \boldsymbol{x}'$ and $\boldsymbol{x} \in I$ then $\boldsymbol{x}' \in I$, and
- (Trajectory closure) For any trajectory $\tau \in \boldsymbol{T}$ if $\tau.fstate \in I$ then $\tau.lstate \in I$

Then I is an inductive invariant of $\mathcal{A}$.

Proof. Consider any reachable state $x \in Reach_{\mathcal{A}}$. By the definition of a reachable state, there exists an execution α of $\mathcal{A}$ with $\alpha.lstate = x$. We proceed by induction on the length of the execution α. For the base case, α consists of a single starting state $\boldsymbol{x} \in \Theta$, and, by the *start condition*, $\boldsymbol{x} \in I$. For the inductive step, we consider two subcases:

Case 1: $\alpha = \alpha' a\, p(\boldsymbol{x})$, where $a \in A$ and $p(\boldsymbol{x})$ is a point trajectory at $\boldsymbol{x}$.

By the induction hypothesis, we know that $\alpha'.lstate \in I$.

By invoking the *transition closure*, we obtain $\boldsymbol{x} \in I$.

Case 2: $\alpha = \alpha' \tau$, where τ is a trajectory of $\mathcal{A}$ and $\tau.lstate = \boldsymbol{x}$

By the *induction hypothesis*, $\alpha'.lstate \in I$ and by

invoking the *trajectory closure*, we deduce that $\tau.lstate = \boldsymbol{x} \in I$

An application

Can you find some candidate invariant for the simple Bouncingball hybrid automaton?

automaton Bouncingball(c,h,g)

variables: x: Reals := h, v: Reals := 0

actions: bounce

transitions:

bounce

pre $x = 0 \wedge v < 0$

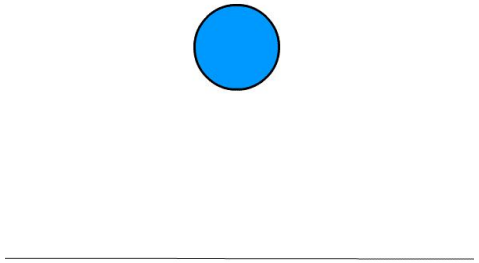
eff v := -cv

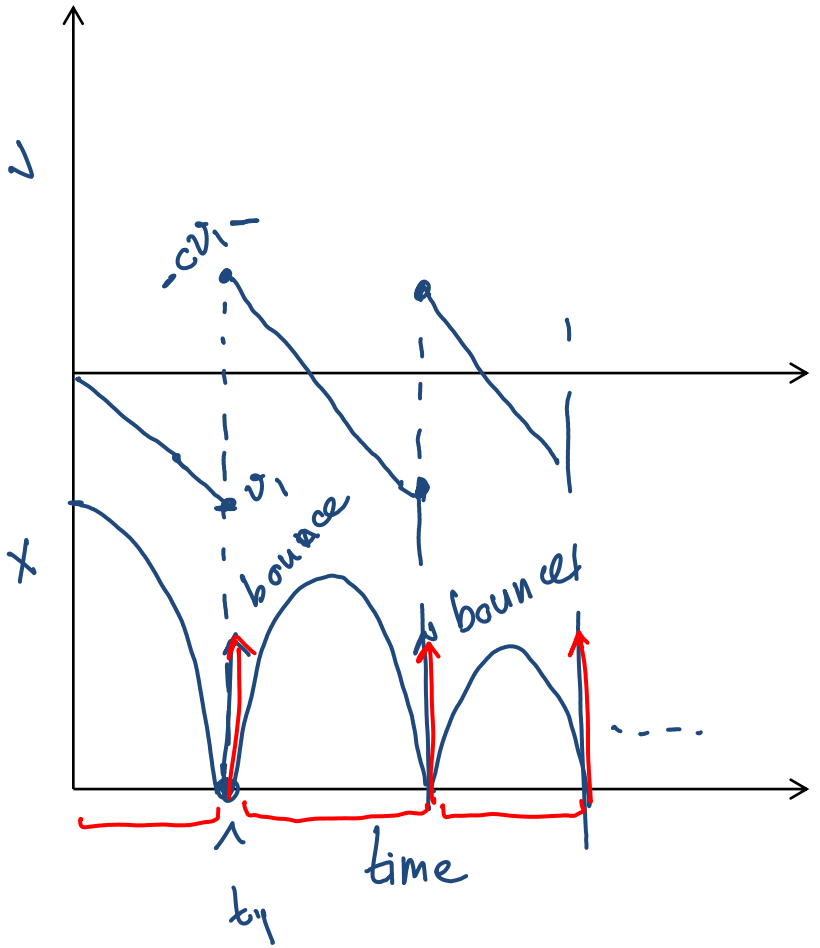
trajectories:

freefall

evolve d(x) = v; d(v) = -g

invariant $x \geq 0$





An application

automaton Bouncingball(c,h,g)

variables: x: Reals := h, v: Reals := 0

actions: bounce

transitions:

bounce

pre *x = 0 /\ v < 0*

eff v := -cv

trajectories:

freefall

evolve d(x) = v; d(v) = -g

invariant $\boldsymbol{x \geq 0}$

Candidate invariant: ``stays above ground''

$I_0 : x \geq 0 \equiv \{\boldsymbol{u} \in val(\{x, v\}) | \ \boldsymbol{u}\lceil x \geq 0 \}$

Applying Theorem 7.1:

- Consider any initial state $\boldsymbol{u} \in \Theta; \boldsymbol{u}\lceil x = h \geq 0$
  - $\boldsymbol{u} \in I_0$
- Consider any transition $\boldsymbol{u} \rightarrow_{bounce} \boldsymbol{u}'$
  - From precondition we know $\boldsymbol{u}\lceil x = 0$; from effect we know $\boldsymbol{u}'.x = \boldsymbol{u}.x$ therefore $\boldsymbol{u}'\lceil x = 0 \geq 0$
  - $\boldsymbol{u}' \in I_0$
- Consider any trajectory $\tau \in T$
  - From mode invariant we know that for $\forall t \in \tau.\ dom\ ,\ \tau(t)\lceil x \geq 0$
  - It follows that $\tau.lstate\lceil x \geq 0$

An application

automaton Bouncingball(c,h,g)

variables: x: Reals := h, v: Reals := 0

actions: bounce

transitions:

bounce

pre *x = 0 /\ v < 0*

eff v := -cv

trajectories:

freefall

evolve d(x) = v; d(v) = -g

invariant $x \geq 0$

Candidate invariant: ``stays above ground and below h''

$I_h: h \geq x \geq 0$

Applying Theorem 7.1:

- Consider any initial state $\boldsymbol{u} \in \Theta$; $\boldsymbol{u}\lceil x = h$
  - $\boldsymbol{u} \in I_h$
- Consider any transition $\boldsymbol{u} \rightarrow_{bounce} \boldsymbol{u}'$
  - From precondition we know $\boldsymbol{u}\lceil x = 0$; from effect we know $\boldsymbol{u}'.x = \boldsymbol{u}.x$ therefore $\boldsymbol{u}'\lceil x = 0$
  - $\boldsymbol{u}' \in I_h$
- Consider any trajectory $\tau \in T$
  - From mode invariant and inductive hypothesis we know that for $\forall t \in \tau.\ dom$, $\tau(t)\lceil x \geq 0$ **and** , $\tau(0)\lceil x \in [0, h]$ and that τ is a solution of d(x) = v; d(v) = -g
  - Is this adequate to infer $\tau.\ lstate \in I_h$?

An application

automaton Bouncingball(c,h,g)

variables: x: Reals := h, v: Reals := 0

actions: bounce

transitions:

bounce

pre *x = 0 /\ v < 0*

eff v := -cv

trajectories:

freefall

evolve d(x) = v; d(v) = -g

invariant $x \geq 0$

Candidate invariant: ``stays above ground and below h''

$I_h: h \geq x \geq 0$

Applying Theorem 7.1:

- Consider any initial state $\boldsymbol{u} \in \Theta$; $\boldsymbol{u}\lceil x = h$
  - $\boldsymbol{u} \in I_h$
- Consider any transition $\boldsymbol{u} \rightarrow_{bounce} \boldsymbol{u'}$
  - From precondition we know $\boldsymbol{u}\lceil x = 0$; from effect we know $\boldsymbol{u'}.x = \boldsymbol{u}.x$ therefore $\boldsymbol{u'}\lceil x = 0$
  - $\boldsymbol{u'} \in I_h$
- Consider any trajectory $\tau \in T$
  - From mode invariant and inductive hypothesis we know that for $\forall t \in \tau.\ dom\ ,\ \tau(t)\lceil x \geq 0$ **and** , $\tau(0)\lceil x \in [0, h]$ and that τ is a solution of d(x) = v; d(v) = -g

No, $I_h: h \geq x \geq 0$ is not an **inductive invariant**! velocity unconstratined. How to fix it?

Strengthened invariant

automaton Bouncingball(c,h,g)

variables: x: Reals := h, v: Reals := 0

k: Nat := 0

actions: bounce

transitions:

bounce

pre *x = 0 /\ v < 0*

eff v := -cv; k:= k + 1

trajectories:

freefall

evolve d(x) = v; d(v) = -g

invariant $x \geq 0$

Candidate invariant: ``stays above ground and below h''

To prove using inductive invariant, strength it!

$I_v: v^2 - 2g(hc^{2k} - x) = 0$

k is the number of bounces capturing the velocity information.

Applying Theorem 7.1:

- Consider any initial state $\boldsymbol{u} \in \Theta$; $\boldsymbol{u}\lceil x = h$; $\boldsymbol{u}\lceil k = 0$
  - $\boldsymbol{u} \in I_v$
- **Exercise**: Finish the rest

Strengthened invariant

automaton Bouncingball(c,h,g)

variables: x: Reals := h, v: Reals := 0

k: Nat := 0

actions: bounce

transitions:

bounce

pre *x = 0 /\ v < 0*

eff v := -cv; k:= k + 1

trajectories:

freefall

evolve d(x) = v; d(v) = -g

invariant $x \geq 0$

Candidate invariant: ``stays above ground and below h''

To prove using inductive invariant, strength it!

$I_v: v^2 - 2g(hc^{2k} - x) = 0$

$I_h: h \geq x \geq 0$

Applying Theorem 7.1 you can prove that $I_v \wedge I_h$ is an (stronger) inductive invariant. So I_h is proved.

Summary: invariants of hybrid automaton

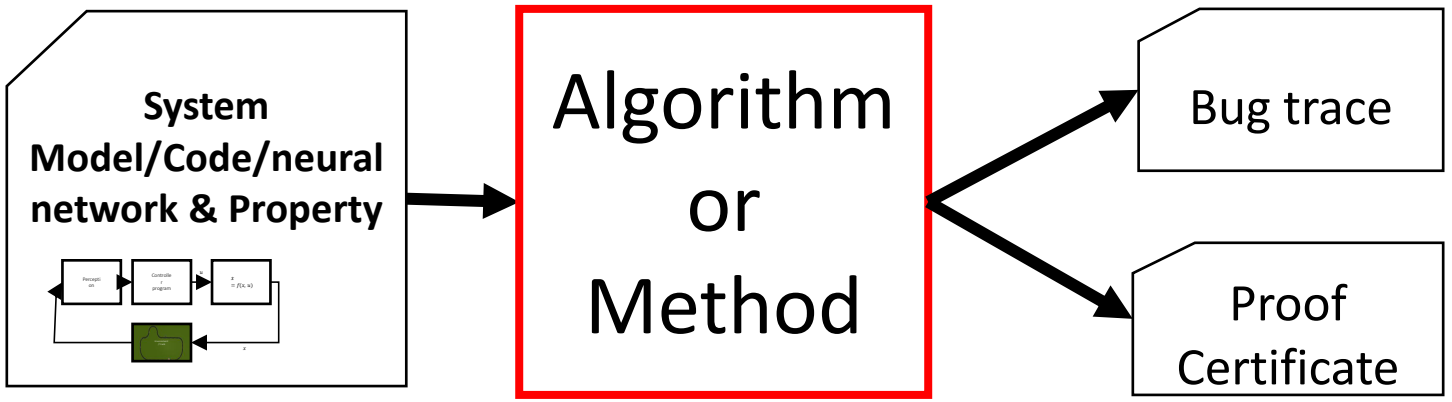
- Theorem 7.1 gives a sufficient condition for proving **inductive** invariants
- Not all invariants are inductive
- We often have to **strengthen** invariants to make them inductive
- Invariants can also be proven using alternative methods (e.g., barrier functions, subangential conditions)
- Read examples in Chapter 7

Back to the verification problem

System. A program/system for *lane keeping control* for vehicles

Model/assumptions: automaton, ODEs, hybrid automata

Requirement. The vehicle does not go outside the lane boundaries



counterexample. A particular environment situation (lane geometry, sensor failure, computer configuration) that makes the vehicle go outside lanes

A mathematical proof that establishes that for all *allowed* inputs and environments the vehicle stays with the lane

Verification tool

When can we build such a tool? How expensive is it? How well is it going to work? Under what assumptions?

Requirements and safety in the real world

Requirements analysis: Set of tasks that ultimately lead to the determination and documentation of the design requirements that the product must meet:

E.g. “0 to 60 mph in 4 seconds on flat road”,

“Petrol car can emit no more than 60mg/km” EURO 6.

Safety standards: Provide *guidelines* and *processes* for developing safety-critical systems.

E.g. DO-178C standard is enforced by the FAA for certifying aviation software

ISO2626 is used for functional safety of cars

Standards for Advanced Autonomous and AI-enabled systems are being developed

THE WHITE HOUSE

OCTOBER 30, 2023

Executive Order on the Safe, Secure, and Trustworthy Development and Use of Artificial Intelligence

BRIEFING ROOM ▸ PRESIDENTIAL ACTIONS

By the authority vested in me as President by the Constitution and the laws of the United States of America, it is hereby ordered as follows:

IEEE SA
STANDARDS ASSOCIATION

IEEE Standard for Robustness Testing and Evaluation of Artificial Intelligence (AI)-based Image Recognition Service

Requirements thus far: Invariants and stability

Models automaton, hybrid automaton $\mathcal{A} = \langle X, \Theta, A, \boldsymbol{\mathcal{D}}, \boldsymbol{T} \rangle$

Requirements: $I \subseteq val(X)$, such that $Reach_{\mathcal{A}}(\Theta) \subseteq I$

Given an **unsafe set** $U \subseteq val(X)$ we can check whether $I \cap U = \emptyset$ to infer that $Reach_{\mathcal{A}}(\Theta) \cap U = \emptyset$

Asymptotic stability: Does $\alpha(\mathrm{x}_0, \mathrm{t}) \to 0$ as $t \to 0$.

What are the requirements that we haven't discussed yet?

What about more general types of requirements, e.g.,

"**Eventually** the light turns red and prior to that the orange light blinks"

"After failures, **eventually** there is just one token in the system"

How to express and verify such properties?

Introduction to temporal logics

Temporal logics: Formal language for representing, and reasoning about, propositions qualified in terms of a **sequence**

Amir Pnueli received the ACM Turing Award (1996) for seminal work introducing temporal logic into computer science and for outstanding contributions to program verification.

Large follow-up literature, e.g., different temporal logics MTL, MITL, PCTL, ACTL, STL, applications in synthesis and monitoring

Setup: States are labeled

We have a set of atomic propositions (AP)

These are the properties that hold in each state, e.g., “light is green”, “has 2 tokens”

We have a *labeling function* that assigns to each state, a set of propositions that hold at that state

$L: Q \rightarrow 2^{AP}$

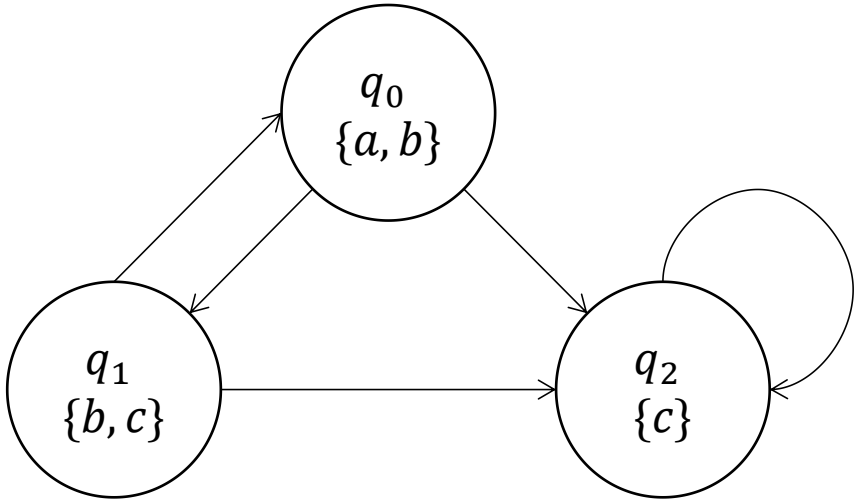
Notations

Automata with state labels but no action labels (“Kripke structure”)

$\mathcal{A} = \langle Q, Q_0, T, L \rangle$

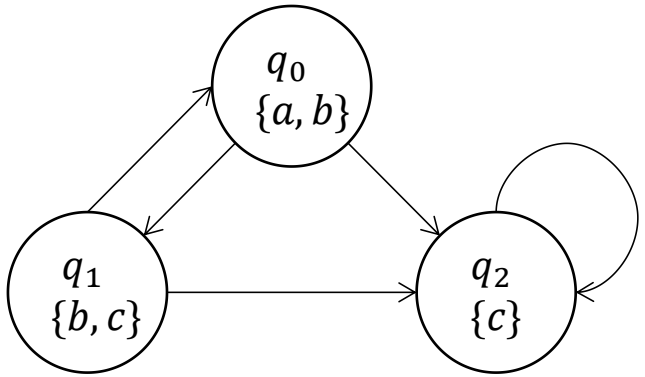
$AP = \{a, b, c\}$

$L(q_0) = \{a, b\}$

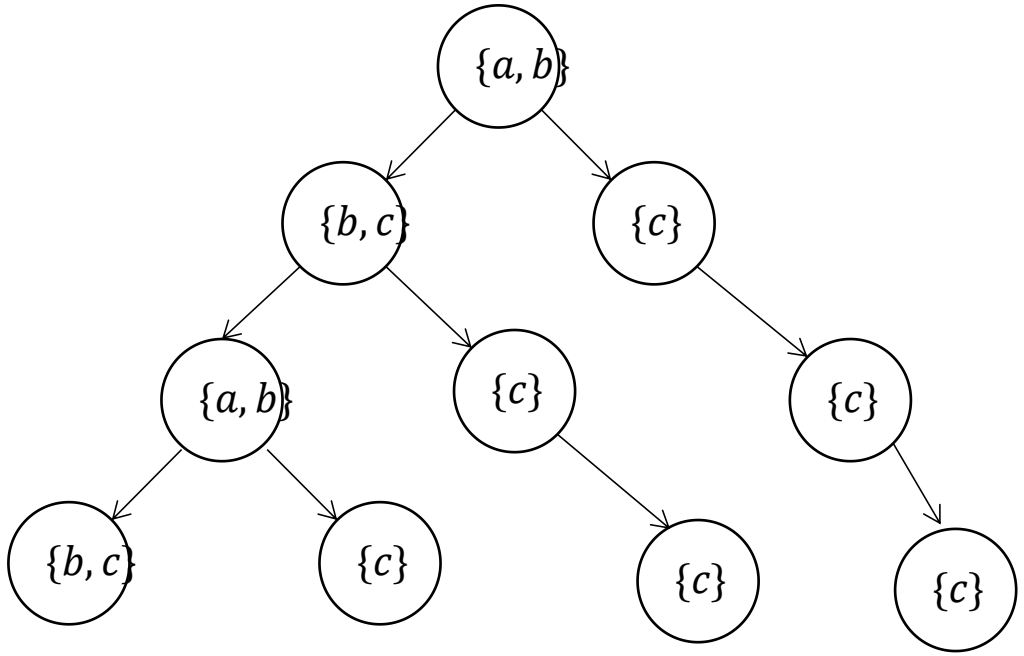


Computational tree logic (CTL)

Unfolding the automaton



We get a tree, representing all possible compuations



A CTL formula allows us to specify subsets of paths in this tree

CTL quantifiers

Path quantifiers

E: Exists some path
A: All paths

Temporal operators

X: Next state
U: Until
F: Eventually
G: Globally (Always)

More details after the Spring break!

ϕ: “no collision”
Invariance: $AG\phi$

ϕ: “one token”
Stabilization: $AF\phi$