Mixed-monotone Theory for Verification of Autonomous System

Saber Jafarpour

University of Colorado Boulder

April 8, 2024
Safety-critical Autonomous Systems

Introduction

Energy/power systems  Air mobility  Autonomous driving
Manufacturing  Transportation systems  Agriculture

An important goal (Safe Autonomy)
Perform their tasks while ensuring safety and robustness of the system.
An important goal (Safe Autonomy)

Perform their tasks while ensuring **safety** and **robustness** of the system.
In this talk: Autonomous systems with learning-enabled components
Learning-enabled Autonomous Systems
Motivations and Success Stories

In this talk: Autonomous systems with learning-enabled components

Machine learning is a driving force for developments in autonomous systems
In this talk: Autonomous systems with learning-enabled components

Machine learning is a driving force for developments in autonomous systems

- availability of data and computation tools
- performance and efficiency
Learning-enabled Autonomous Systems
Motivations and Success Stories

In this talk: Autonomous systems with learning-enabled components

Machine learning is a driving force for developments in autonomous systems

- availability of data and computation tools
- performance and efficiency

Success stories and potential applications

- NVIDIA self-driving car
- Amazon fulfillment centers
- Manufacturing
Learning-enabled Autonomous Systems
Safety Assurance as a Challenge

But can we ensure their safety?

Tesla Slams Right Into Overturned Truck While on Autopilot

Robot accident at Amazon warehouse renews safety debate

Waymo driverless car strikes bicyclist in San Francisco, causes minor injuries
Learning-enabled Autonomous Systems
Safety Assurance as a Challenge

But can we ensure their safety?

What is different with Learning-based components?

Robot accident at Amazon warehouse renews safety debate

Waymo driverless car strikes bicyclist in San Francisco, causes minor injuries

Tesla Slams Right Into Overturned Truck While on Autopilot

S. Jafarpour (CU Boulder)
But can we ensure their safety?

- limited guarantee in their design
But can we ensure their safety?

- limited guarantee in their design

Robot accident at Amazon warehouse renews safety debate

Waymo driverless car strikes bicyclist in San Francisco, causes minor injuries

- The way we train AI is fundamentally flawed

The process used to build most of the machine-learning models we use today can’t tell if they will work in the real world or not—and that’s a problem.

By Will Douglas Heaven
November 18, 2020
Learning-enabled Autonomous Systems
Safety Assurance as a Challenge

But can we ensure their safety?

- limited guarantee in their design
- large # of parameters with nonlinearity

Robot accident at Amazon warehouse renews safety debate
Waymo driverless car strikes bicyclist in San Francisco, causes minor injuries

- Feedforward neural network
- Implicit neural network

\[
\begin{align*}
\text{# of parameters} & \approx 90000 \\
\text{# of activation patterns} & \approx 10^{60}
\end{align*}
\]
Learning-enabled Autonomous Systems
Safety Assurance as a Challenge

But can we ensure their safety?

- limited guarantee in their design
- large # of parameters with nonlinearity

Rigorous and computationally efficient methods for safety assurance
ML focus on safety and robustness of **stand-alone** learning algorithms

\[ x \rightarrow N(x) \]
ML focus on safety and robustness of **stand-alone** learning algorithms

Different approaches:

- analysis (Goodfellow et al., 2015, Zhang et al., 2019, Fazlyab et al., 2023)
- design (Papernot et al., 2016, Carlini and Wagner, 2017, Madry et al., 2018)
ML focus on safety and robustness of **stand-alone** learning algorithms

Different approaches:

- analysis \((\text{Goodfellow et al., 2015, Zhang et al., 2019, Fazlyab et al., 2023})\)
- design \((\text{Papernot et al., 2016, Carlini and Wagner, 2017, Madry et al., 2018})\)

In autonomous systems, learning algorithms are a part of the system (controller, motion planner, obstacle detection)
ML focus on safety and robustness of **stand-alone** learning algorithms

Different approaches:

- analysis (Goodfellow et al., 2015, Zhang et al., 2019, Fazlyab et al., 2023)
- design (Papernot et al., 2016, Carlini and Wagner, 2017, Madry et al., 2018)

In autonomous systems, learning algorithms are **a part of the system** (controller, motion planner, obstacle detection)

New challenges arises when learning algorithms are used **in-the-loop**
Example: Safety in Mobile Robots

In-the-loop vs. stand-alone

Perception-based Obstacle Avoidance

![Diagram of a system with components such as Disturbance, Actuator, Learning-based obstacle detection, Camera Images, and the equations $u = K(\hat{x})$ and $\hat{x}$.]

In-the-loop:
- How the autonomous system performs with the learning algorithm as a part of it.

Stand-alone:
- Estimation of states using a learning algorithm.
- Trained offline using images.

Learning-based obstacle detection:
- Trained offline using images.
Perception-based Obstacle Avoidance

In-the-loop vs. stand-alone

- **In-the-loop**: How the autonomous system performs with the learning algorithm as a part of it.
- **Stand-alone**: Estimation of states using learning algorithm

- **stand-alone**: Estimation of states using learning algorithm
- **in-the-loop**: Closed-loop system avoids the obstacle

Trained offline using images
Perception-based Obstacle Avoidance

**In-the-loop**

- **stand-alone**: estimation of states using learning algorithm
- **in-the-loop**: closed-loop system avoid the obstacle

**Stand-alone**

- **In-the-loop**: how the autonomous system perform with the learning algorithm as a part of it.
Ensure safety of the autonomous system with learning algorithms in-the-loop
Ensure safety of the autonomous system with learning algorithms **in-the-loop**

Safety of autonomous system using **reachability analysis**
Ensure safety of the autonomous system with learning algorithms *in-the-loop*

Safety of autonomous system using *reachability analysis*

Reachability analysis estimates the evolution of the autonomous system
Ensure safety of the autonomous system with learning algorithms in-the-loop

Safety of autonomous system using reachability analysis

Reachability analysis estimates the evolution of the autonomous system

In this talk:
1. control-theoretic tools for efficient and scalable reachability
2. applications to safety assurance of learning-enabled systems
Outline of this talk

- Reachability Analysis
- Monotone System Theory
- Neural Network Controlled Systems
Problem Statement

**System**: $\dot{x} = f(x, w)$  

**State**: $x \in \mathbb{R}^n$  

**Uncertainty**: $w \in \mathcal{W} \subseteq \mathbb{R}^m$

What are the possible states of the system at time $T$?
Reachability Analysis of Systems

Problem Statement

**System**: \( \dot{x} = f(x, w) \)

**State**: \( x \in \mathbb{R}^n \)

**Uncertainty**: \( w \in \mathcal{W} \subseteq \mathbb{R}^m \)

---

What are the possible states of the system at time \( T \)?

- **\( T \)-reachable sets** characterize evolution of the system

\[
\mathcal{R}_f(T, X_0, \mathcal{W}) = \{ x_w(T) \mid x_w(\cdot) \text{ is a traj for some } w(\cdot) \in \mathcal{W} \text{ with } x_0 \in X_0 \}
\]
A large number of **safety specifications** can be represented using $T$-reachable sets.
A large number of safety specifications can be represented using $T$-reachable sets.

Example: Reach-avoid problem

\[ R_f(T, X_0, \mathcal{W}) \cap \text{Unsafe set} = \emptyset \]

\[ R_f(T_{\text{final}}, X_0, \mathcal{W}) \subseteq \text{Target set} \]
A large number of safety specifications can be represented using $T$-reachable sets.

Example: Reach-avoid problem

\[ \mathcal{R}_f(T, \mathcal{X}_0, \mathcal{W}) \cap \text{Unsafe set} = \emptyset \]

\[ \mathcal{R}_f(T_{\text{final}}, \mathcal{X}_0, \mathcal{W}) \subseteq \text{Target set} \]

Combining different instantiation of Reach-avoid problem $\implies$ diverse range of specifications

(complex planning using logics, invariance, stability)
Computing the $T$-reachable sets are computationally challenging.
Reachability Analysis of Systems

Why is it difficult?

Computing the $T$-reachable sets are computationally challenging

Solution: over-approximations of reachable sets

Over-approximation: $\mathcal{R}_f(T, X_0, \mathcal{W}) \subseteq \overline{\mathcal{R}}_f(T, X_0, \mathcal{W})$
Reachability Analysis of Systems

Why is it difficult?

Computing the $T$-reachable sets are computationally challenging

**Solution:** over-approximations of reachable sets

**Over-approximation:**

$$\overline{R}_f(T, X_0, W) \subseteq \overline{R}_f(T, X_0, W)$$

\[
\overline{R}_f(T, X_0, W) \cap \text{Unsafe set} = \emptyset
\]

\[
\overline{R}_f(T_{\text{final}}, X_0, W) \subseteq \text{Target set}
\]
Reachability Analysis of Systems

Applications

Autonomous Driving:

Althoff, 2014

Robot-assisted Surgery:

Chen, Dutta, and Sankaranarayanan, 2017

Power grids:

Chen and Domínguez-Garcia, 2016

Drug Delivery:
Reachability Analysis of Systems

Literature review

Reachability of dynamical system is an old problem: $\sim 1980$
Reachability of dynamical system is an old problem: \( \sim 1980 \)

Different approaches for approximating reachable sets

- Linear, and piecewise linear systems (Ellipsoidal methods) ([Kurzhanski and Varaiya, 2000](#))
- Optimization-based approaches (Hamilton-Jacobi, Level-set method) ([Bansal et al., 2017](#), [Mitchell et al., 2002](#), [Herbert et al., 2021](#))
- Matrix measure-based ([Fan et al., 2018](#), [Maidens and Arcak, 2015](#))
Reachability Analysis of Systems

Literature review

Reachability of dynamical system is an old problem: \(\sim 1980\)

Different approaches for approximating reachable sets

- Linear, and piecewise linear systems (Ellipsoidal methods) (Kurzhanski and Varaiya, 2000)
- Optimization-based approaches (Hamilton-Jacobi, Level-set method) (Bansal et al., 2017, Mitchell et al., 2002, Herbert et al., 2021)
- Matrix measure-based (Fan et al., 2018, Maidens and Arcak, 2015)

Most of the classical reachability approaches are computationally heavy and not scalable to large-size systems
Reachability Analysis of Systems

Literature review

Reachability of dynamical system is an old problem: \( \sim 1980 \)

Different approaches for approximating reachable sets

- Linear, and piecewise linear systems (Ellipsoidal methods) (Kurzhanski and Varaiya, 2000)
- Optimization-based approaches (Hamilton-Jacobi, Level-set method) (Bansal et al., 2017, Mitchell et al., 2002, Herbert et al., 2021)
- Matrix measure-based (Fan et al., 2018, Maidens and Arcak, 2015)

Most of the classical reachability approaches are computationally heavy and not scalable to large-size systems

In this talk: use control-theoretic tools to develop scalable and computationally efficient approaches for reachability
Outline of this talk

- Reachability Analysis
- **Monotone System Theory**
- Neural Network Controlled Systems
A dynamical system $\dot{x} = f(x, w)$ is monotone if

$$x_u(0) \leq y_w(0) \quad \text{and} \quad u \leq w \implies x_u(t) \leq y_w(t) \quad \text{for all time}$$

where $\leq$ is the component-wise partial order.

---

1Angeli and Sontag, “Monotone control systems”, IEEE TAC, 2003
A dynamical system $\dot{x} = f(x, w)$ is monotone if

$$x_u(0) \leq y_w(0) \quad \text{and} \quad u \leq w \implies x_u(t) \leq y_w(t) \quad \text{for all time}$$

where $\leq$ is the component-wise partial order.

**Theorem**: Monotonicity test

1. $\frac{\partial f}{\partial x}(x, w)$ is Metzler (off-diag $\geq 0$)
2. $\frac{\partial f}{\partial w}(x, w) \geq 0$

---

A dynamical system \( \dot{x} = f(x, w) \) is monotone if

\[
x_u(0) \leq y_w(0) \quad \text{and} \quad u \leq w \quad \implies \quad x_u(t) \leq y_w(t) \quad \text{for all time}
\]

where \( \leq \) is the component-wise partial order.

**Theorem\(^1\): Monotonicity test**

1. \( \frac{\partial f}{\partial x}(x, w) \) is Metzler (off-diag \( \geq 0 \))
2. \( \frac{\partial f}{\partial w}(x, w) \geq 0 \)

---

\(^1\) Angeli and Sontag, “Monotone control systems”, IEEE TAC, 2003

**In this talk**: monotone system theory for reachability analysis
Monotone System

\[
\frac{d}{dt} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} x_2^3 - x_1 + w \\ x_1 \end{bmatrix}
\]

Non-monotone System

\[
\frac{d}{dt} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} x_2^3 - x_2 + w \\ x_1 \end{bmatrix}
\]
Theorem (classical result)

For a monotone system $\dot{x} = f(x, w)$ with $w \in \mathcal{W} = [w, \bar{w}]

\[ R_f(t, [x_0, \bar{x_0}], [w, \bar{w}]) \subseteq [x_w(t), x_{\bar{w}}(t)] \]

where $x_w(\cdot)$ (resp. $x_{\bar{w}}(\cdot)$) is the trajectory with disturbance $w$ (resp. $\bar{w}$) starting at $x_0$ (resp. $\bar{x_0}$)
**Theorem (classical result)**

For a monotone system \( \dot{x} = f(x, w) \) with \( w \in \mathcal{W} = [w, \bar{w}] \)

\[
\mathcal{R}_f(t, [x_0, \bar{x}_0], [w, \bar{w}]) \subseteq [x_w(t), x_{\bar{w}}(t)]
\]

where \( x_w(\cdot) \) (resp. \( x_{\bar{w}}(\cdot) \)) is the trajectory with disturbance \( w \) (resp. \( \bar{w} \)) starting at \( x_0 \) (resp. \( \bar{x}_0 \))

**Example:**

\[
\frac{d}{dt} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} x_2^3 - x_1 + w \\ x_1 \end{bmatrix}
\]

\( \mathcal{W} = [2.2, 2.3] \) \( \mathcal{X}_0 = \left[ \begin{bmatrix} -0.5 \\ -0.5 \end{bmatrix}, \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix} \right] \)
A large number of the dynamical systems are **not** monotone.
A large number of the dynamical systems are not monotone.

For non-monotone dynamical systems, the extreme trajectories do not provide any over-approximation of reachable sets.
A large number of the dynamical systems are not monotone. For non-monotone dynamical systems the extreme trajectories do not provide any over-approximation of reachable sets.

Example:

\[
\frac{d}{dt} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} x_2^3 - x_2 + w \\ x_1 \end{bmatrix}
\]

\[\mathcal{W} = [2.2, 2.3] \quad \mathcal{X}_0 = \left[ \begin{bmatrix} -0.5 \\ -0.5 \end{bmatrix}, \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix} \right] \]

\[
\begin{array}{cc}
\text{Not Overapproximation} \\
\end{array}
\]
A large number of the dynamical systems are not monotone.

For non-monotone dynamical systems, the extreme trajectories do not provide any over-approximation of reachable sets.

Example:

\[
\frac{dx_1}{dt} = x_3^2 - x_2 + w \\
\frac{dx_2}{dt} = x_1
\]

\[W = [2.2, 2.3] \quad x_0 = \left[-0.5, 0.5\right] \cup \left[0.5, 0.5\right]\]

How to over-approximate the reachable sets of non-monotone systems?
**Key idea:** embed the dynamical system on $\mathbb{R}^n$ into a dynamical system on $\mathbb{R}^{2n}$

Assume $\mathcal{W} = [w, \bar{w}]$ and $\mathcal{X}_0 = [x_0, \bar{x}_0]$

Original system

\[
\dot{x} = f(x, w)
\]

Embedding system

\[
\begin{align*}
\dot{x} &= d(x, \bar{x}, w, \bar{w}), \\
\dot{\bar{x}} &= \bar{d}(x, \bar{x}, w, \bar{w})
\end{align*}
\]

$d, \bar{d}$ are decomposition functions s.t.

1. $f(x, w) = d(x, x, w, w)$ for every $x, w$
2. cooperative: $(x, w) \mapsto d(x, \bar{x}, w, \bar{w})$
3. competitive: $(\bar{x}, \bar{w}) \mapsto \bar{d}(x, \bar{x}, w, \bar{w})$
4. the same properties for $\bar{d}$
**Key idea:** embed the dynamical system on $\mathbb{R}^n$ into a dynamical system on $\mathbb{R}^{2n}$

- Assume $\mathcal{W} = [w, \bar{w}]$ and $\mathcal{X}_0 = [x_0, \bar{x}_0]$

**Original system**

$$\dot{x} = f(x, w)$$

**Embedding system**

$$\dot{x} = d(x, \bar{x}, w, \bar{w}),$$

$$\dot{x} = \bar{d}(x, \bar{x}, w, \bar{w})$$

$d, \bar{d}$ are **decomposition functions** s.t.

1. $f(x, w) = d(x, x, w, w)$ for every $x, w$
2. cooperative: $(x, w) \mapsto d(x, x, w, w)$
3. competitive: $(\bar{x}, \bar{w}) \mapsto d(x, \bar{x}, w, \bar{w})$
4. the same properties for $\bar{d}$

$f$ locally Lipschitz $\implies$ a decomposition function exists
Southeast partial order \( \leq_{SE} \):
\[
\begin{bmatrix} x \\ \hat{x} \end{bmatrix} \leq_{SE} \begin{bmatrix} y \\ \hat{y} \end{bmatrix} \iff x \leq y \quad \text{and} \quad \hat{y} \leq \hat{x}
\]
**Southeast partial order** \(\leq_{SE}\):

\[
\begin{bmatrix} x \\ \hat{x} \end{bmatrix} \leq_{SE} \begin{bmatrix} y \\ \hat{y} \end{bmatrix} \iff x \leq y \text{ and } \hat{y} \leq \hat{x}
\]

**Theorem (Classical Result)**

The embedding system is a monotone dynamical system on \(\mathbb{R}^{2n}\) with respect to the *southeast* partial order \(\leq_{SE}\):

\[
\begin{bmatrix} x_0 \\ \bar{x}_0 \end{bmatrix} \leq_{SE} \begin{bmatrix} y_0 \\ \bar{y}_0 \end{bmatrix}, \quad \begin{bmatrix} u \\ \bar{u} \end{bmatrix} \leq_{SE} \begin{bmatrix} w \\ \bar{w} \end{bmatrix} \implies \begin{bmatrix} x_{[u,\bar{u}]}(t) \\ \bar{x}_{[u,\bar{u}]}(t) \end{bmatrix} \leq_{SE} \begin{bmatrix} y_{[w,\bar{w}]}(t) \\ \bar{y}_{[w,\bar{w}]}(t) \end{bmatrix}
\]
Mixed Monotone Theory
Southeast partial order on $\mathbb{R}^{2n}$

**Southeast** partial order $\leq_{SE}$:

$$\begin{bmatrix} x \\ \hat{x} \end{bmatrix} \leq_{SE} \begin{bmatrix} y \\ \hat{y} \end{bmatrix} \iff x \leq y \text{ and } \hat{y} \leq \hat{x}$$

**Theorem (Classical Result)**

The embedding system is a monotone dynamical system on $\mathbb{R}^{2n}$ with respect to the **southeast** partial order $\leq_{SE}$:

$$\begin{bmatrix} x_0 \\ \bar{x}_0 \end{bmatrix} \leq_{SE} \begin{bmatrix} y_0 \\ \bar{y}_0 \end{bmatrix}, \quad \begin{bmatrix} u \\ \bar{u} \end{bmatrix} \leq_{SE} \begin{bmatrix} w \\ \bar{w} \end{bmatrix} \implies \begin{bmatrix} x_{[u,\bar{u}]}(t) \\ \bar{x}_{[u,\bar{u}]}(t) \end{bmatrix} \leq_{SE} \begin{bmatrix} y_{[w,\bar{w}]}(t) \\ \bar{y}_{[w,\bar{w}]}(t) \end{bmatrix}$$

**Key idea:** use monotonicity of the embedding system to study the original dynamical system
A short (and incomplete) Literature review:


A short (and incomplete) Literature review:


In this talk: use embedding system to study reachability of the original system
Original System:

\[
\begin{align*}
\frac{d}{dt} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} &= \begin{bmatrix} x_2^3 - x_2 + w \\ x_1 \end{bmatrix} \\
W &= [2.2, 2.3]
\end{align*}
\]

blue = cooperative, red = competitive

Decomposition function

\[
\begin{align*}
d(x, \bar{x}, w, \bar{w}) &= \begin{bmatrix} x_2^3 + w \\ x_1 \end{bmatrix} + \begin{bmatrix} -\bar{x}_2 \\ 0 \end{bmatrix} \\
\bar{d}(x, \bar{x}, w, \bar{w}) &= \begin{bmatrix} \bar{x}_2^3 + \bar{w} \\ \bar{x}_1 \end{bmatrix} + \begin{bmatrix} -\bar{x}_2 \\ 0 \end{bmatrix}
\end{align*}
\]
**Original System:**

\[
\begin{align*}
\frac{d}{dt} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} &= \begin{bmatrix} x_2^3 - x_2 + w \\ x_1 \end{bmatrix} \\
\mathcal{W} &= [2.2, 2.3]
\end{align*}
\]

blue = cooperative, red = competitive

**Embedding System:**

\[
\begin{align*}
\frac{d}{dt} \begin{bmatrix} x_1 \\ x_2 \\ \overline{x}_1 \\ \overline{x}_2 \end{bmatrix} &= \begin{bmatrix} x_2^3 - \overline{x}_2 + w \\ x_1 \\ \overline{x}_2^3 - \overline{x}_2 + w \\ \overline{x}_1 \end{bmatrix} \\
\begin{bmatrix} w \\ \overline{w} \end{bmatrix} &= \begin{bmatrix} 2.2 \\ 2.3 \end{bmatrix}
\end{align*}
\]

**Decomposition function**

\[
\begin{align*}
\mathcal{d}(x, \overline{x}, w, \overline{w}) &= \begin{bmatrix} x_2^3 + w \\ x_2 \\ x_1 \end{bmatrix} + \begin{bmatrix} -\overline{x}_2 \\ 0 \end{bmatrix} \\
\overline{\mathcal{d}}(x, \overline{x}, w, \overline{w}) &= \begin{bmatrix} \overline{x}_2^3 + \overline{w} \\ \overline{x}_2 \\ \overline{x}_1 \end{bmatrix} + \begin{bmatrix} -\overline{x}_2 \\ 0 \end{bmatrix}
\end{align*}
\]
Linear Dynamical System

A structure preserving decomposition function

- **Metzler/non-Metzler decomposition**: \( A = [A]^{Mzl} + [A]^{n-Mzl} \)

- **Example**: \( A = \begin{bmatrix} 2 & 0 & -1 \\ 1 & -3 & 0 \\ 0 & 0 & 1 \end{bmatrix} \Rightarrow [A]^{Mzl} = \begin{bmatrix} 2 & 0 & 0 \\ 1 & -3 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad [A]^{n-Mzl} = \begin{bmatrix} 0 & 0 & -1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \)

Linear systems

Original system

\[
\dot{x} = Ax + Bw
\]

Embedding system

\[
\dot{x} = [A]^{Mzl} x + [A]^{n-Mzl} \bar{x} + B^+ w + B^- w
\]

\[
\dot{x} = [A]^{Mzl} \bar{x} + [A]^{n-Mzl} x + B^+ \bar{w} + B^- w
\]
Decomposition Functions
A Jacobian-based approach

How to compute a decomposition function for a system?

- Assume $f : \mathbb{R} \rightarrow \mathbb{R}$ is scalar-valued:

Mean-value Inequality

\[
\begin{align*}
    f(x) + \left[ \min_{z \in [x, \bar{x}]} \frac{\partial f}{\partial x} \right] (\bar{x} - x) & \leq f(x) \leq f(x) + \left[ \max_{z \in [x, \bar{x}]} \frac{\partial f}{\partial x} \right] (\bar{x} - x) \\
\end{align*}
\]

Then

\[
\begin{align*}
    \begin{bmatrix}
        d(x, \bar{x}) \\
        d(x, \bar{x})
    \end{bmatrix} = 
    \begin{bmatrix}
        \left[ \min_{z \in [x, \bar{x}]} \frac{\partial f}{\partial x} \right]^+ \\
        \left[ \max_{z \in [x, \bar{x}]} \frac{\partial f}{\partial x} \right]^-
    \end{bmatrix} 
    \begin{bmatrix}
        x \\
        \bar{x}
    \end{bmatrix}
\end{align*}
\]

where $[A]^+ = \max\{A, 0\}$ and $[A]^− = \min\{A, 0\}$.
Decomposition Functions
A Jacobian-based approach

How to compute a decomposition function for a system?

**Theorem**

**Jacobian-based:** \( \dot{x} = f(x, w) \) with differentiable \( f \), then

\[
\frac{d(x, \bar{x}, u, \bar{u})}{d(x, \bar{x}, w, \bar{w})} = \begin{bmatrix} [A]^+ & [A]^- \end{bmatrix} \begin{bmatrix} x \end{bmatrix} + \begin{bmatrix} [B]^+ & [B]^- \end{bmatrix} \begin{bmatrix} w \end{bmatrix} + \begin{bmatrix} f(x, w) \end{bmatrix}
\]

\( x \mapsto R_1 \mapsto R_2 \mapsto \ldots \mapsto R_n \mapsto \bar{x} \), then the \( i \)-th column of \( A \) is \( \min_{z \in R_i, u \in [w, \bar{w}]} \frac{\partial f_i}{\partial x}(z, u) \)

- Interval analysis for computing Jacobian bounds.
- immrax: Toolbox that implements interval analysis in JAX.

\(^2\text{SJ and A. Harapanahalli and S. Coogan, L4DC, 2023}\)
Decomposition Functions
A Jacobian-based approach

How to compute a decomposition function for a system?

**Theorem**

**Jacobian-based:** \( \dot{x} = f(x, w) \) with differentiable \( f \), then

\[
\frac{d(x, \bar{x}, u, \bar{u})}{d(x, \bar{x}, w, \bar{w})} = \left[ \frac{[A]^+}{[A]^+} \frac{[A]^-}{[A]^+} \right] \frac{x}{\bar{x}} + \left[ \frac{[B]^+}{[B]^+} \frac{[B]^-}{[B]^+} \right] \frac{w}{\bar{w}} + \frac{f(x, w)}{f(x, w)}
\]

\( x \mapsto R_1 \mapsto R_2 \mapsto \ldots \mapsto R_n \mapsto \bar{x} \), then the \( i \)-th column of \( A \) is \( \min_{z \in R_i, u \in [w, \bar{w}]} \frac{\partial f_i}{\partial x} (z, u) \)

- Interval analysis for computing Jacobian bounds.
- **immrax:** Toolbox that implements interval analysis in JAX.

\(^2\)SJ and A. Harapanahalli and S. Coogan, L4DC, 2023
Interval-based Reachability
Embedding Systems

Theorem

Assume $\mathcal{W} = [\underline{w}, \overline{w}]$ and $\mathcal{X}_0 = [\underline{x}_0, \overline{x}_0]$ and

$$\begin{align*}
\dot{x} &= d(x, \overline{x}, w, \overline{w}), \quad x(0) = x_0 \\
\dot{\overline{x}} &= \overline{d}(\overline{x}, x, w, w), \quad \overline{x}(0) = \overline{x}_0
\end{align*}$$

Then $R_f(t, \mathcal{X}_0, \mathcal{W}) \subseteq [\underline{x}(t), \overline{x}(t)]$

---

Theorem\textsuperscript{3}

Assume $\mathcal{W} = [w, \bar{w}]$ and $\mathcal{X}_0 = [x_0, \bar{x}_0]$ and

\[
\dot{x} = d(x, \bar{x}, w, \bar{w}), \quad x(0) = x_0
\]
\[
\dot{\bar{x}} = \bar{d}(\bar{x}, x, w, \bar{w}), \quad \bar{x}(0) = \bar{x}_0
\]

Then $\mathcal{R}_f(t, \mathcal{X}_0, \mathcal{W}) \subseteq [x(t), \bar{x}(t)]$

a single trajectory of embedding system provides lower bound $\underline{x}$ and upper bound $\overline{x}$ for the trajectories of the original system.

---

Theorem

Assume $W = [w, \bar{w}]$ and $X_0 = [x_0, \bar{x}_0]$ and

$$\dot{x} = d(x, \bar{x}, w, \bar{w}), \quad x(0) = x_0$$
$$\dot{x} = \overline{d}(\bar{x}, x, \bar{w}, w), \quad \bar{x}(0) = \bar{x}_0$$

Then $R_f(t, X_0, W) \subseteq [x(t), \bar{x}(t)]$

a single trajectory of embedding system provides lower bound ($\underline{x}$) and upper bound ($\overline{x}$) for the trajectories of the original system.

(Computational efficient): solve for one trajectory of embedding system
(Scalable): embedding system is $2n$-dimensional

---

Original System:

\[
\begin{bmatrix}
\frac{dx_1}{dt} \\
\frac{dx_2}{dt}
\end{bmatrix} = \begin{bmatrix}
x_2^3 - x_2 + w \\
x_1
\end{bmatrix}
\]

\[\mathcal{W} = [2.2, 2.3] \quad \mathcal{X}_0 = \begin{bmatrix}
\begin{bmatrix} -0.5 \end{bmatrix}, \\
\begin{bmatrix} 0.5 \end{bmatrix}
\end{bmatrix}
\]

blue = cooperative, red = competitive

Decomposition function

\[
d(x, \bar{x}, w, \bar{w}) = \begin{bmatrix}
x_2^3 + w \\
-x_2
\end{bmatrix} + \begin{bmatrix}
-x_2 \\
0
\end{bmatrix}
\]

\[
\bar{d}(x, \bar{x}, w, \bar{w}) = \begin{bmatrix}
x_2^3 + w \\
-x_2
\end{bmatrix} + \begin{bmatrix}
-x_2 \\
0
\end{bmatrix}
\]
Reachability using Embedding Systems

**Example**

**Original System:**

\[
\frac{d}{dt} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} x_2^3 - x_2 + w \\ x_1 \end{bmatrix}
\]

\[\mathcal{W} = [2.2, 2.3] \quad \mathcal{X}_0 = \begin{bmatrix} -0.5 \\ -0.5 \end{bmatrix}, \begin{bmatrix} 0.5 \end{bmatrix}\]

blue = cooperative, red = competitive

**Embedding System:**

\[
\frac{d}{dt} \begin{bmatrix} x_1 \\ x_2 \\ x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} x_2^3 - x_2 + w \\ x_1 \\ x_2^3 - x_2 + w \\ x_1 \end{bmatrix}
\]

\[\mathcal{W} = [2.2, 2.3] \quad \mathcal{X}_0 = \begin{bmatrix} -0.5 \\ -0.5 \end{bmatrix}, \begin{bmatrix} 0.5 \end{bmatrix}\]

**Decomposition function**

\[
d(x, \bar{x}, w, \bar{w}) = \begin{bmatrix} x_2^3 + w \\ x_1 \end{bmatrix} + \begin{bmatrix} -x_2 \\ 0 \end{bmatrix}
\]

\[
\bar{d}(x, \bar{x}, w, \bar{w}) = \begin{bmatrix} x_2^3 + w \\ x_1 \end{bmatrix} + \begin{bmatrix} -x_2 \\ 0 \end{bmatrix}
\]
Outline of this talk

- Reachability Analysis
- Monotone System Theory
- Neural Network Controlled Systems
Learning-based Controllers in Autonomous Systems

Introduction

In this part: Learning-based component as a controller
In this part: Learning-based component as a controller

- Computationally burdensome
- Interaction with human
- Complicated representation

Learning-based Feedback

System

Disturbance

In this part: Learning-based component as a controller

Issues with traditional controllers:
1. computationally burdensome
2. interaction with human
3. complicated representation

Self driving vehicles:

Robotic motion planning:
M. Everett, et. al., IROS, 2018.

Collision avoidance:
Safety of learning-enabled autonomous systems cannot be completely ensured at the design level\textsuperscript{4}.

\textsuperscript{4}Institute for Defense Analysis, The Status of Test, Evaluation, Verification, and Validation of Autonomous Systems, 2018
Safety of learning-enabled autonomous systems cannot be completely ensured at the design level\textsuperscript{4}

---

\textsuperscript{4}Institute for Defense Analysis, The Status of Test, Evaluation, Verification, and Validation of Autonomous Systems, 2018
Safety of learning-enabled autonomous systems cannot be completely ensured at the design level\textsuperscript{4}.

Design a mechanism that can do \textit{run-time} safety verification.

\textsuperscript{4}Institute for Defense Analysis, The Status of Test, Evaluation, Verification, and Validation of Autonomous Systems, 2018
Safety of learning-enabled autonomous systems cannot be completely ensured at the design level\(^4\).

Design a mechanism that can do run-time safety verification.

**Our approach**: reachable set over-approximations for some time in future.

\(^4\text{Institute for Defense Analysis, The Status of Test, Evaluation, Verification, and Validation of Autonomous Systems, 2018}\)
An open-loop nonlinear system with a neural network controller

\[ \dot{x} = f(x, u, w), \]
\[ u = N(x), \]

safety of the closed-loop system

\[ \dot{x} = f(x, N(x), w) := f^c(x, w) \]
An open-loop nonlinear system with a neural network controller

\[
\dot{x} = f(x, u, w), \\
u = N(x),
\]
safety of the closed-loop system

\[
\dot{x} = f(x, N(x), w) := f^c(x, w)
\]

\(u = N(x)\) is \textit{pre-trained} feed-forward neural network with \(k\)-layer:

\[
\xi^{(i)}(x) = \phi^{(i)}(W^{(i-1)}\xi^{(i-1)}(x) + b^{(i-1)})
\]
\(x = \xi^{(0)}, u = W^{(k)}\xi^{(k)}(x) + b^{(k)} := N(x),\)
An open-loop nonlinear system with a neural network controller

\[
\dot{x} = f(x, u, w),
\]
\[
u = N(x),
\]
safety of the closed-loop system

\[
\dot{x} = f(x, N(x), w) := f^c(x, w)
\]

\(u = N(x)\) is **pre-trained** feed-forward neural network with \(k\)-layer:

\[
\xi^{(i)}(x) = \phi^{(i)}(W^{(i-1)}\xi^{(i-1)}(x) + b^{(i-1)})
\]
\[
x = \xi^{(0)}, \quad u = W^{(k)}\xi^{(k)}(x) + b^{(k)} := N(x),
\]

directly performing reachability on \(f^c\) is computationally challenging
An open-loop nonlinear system with a neural network controller
\[
\dot{x} = f(x, u, w), \quad u = N(x),
\]
safety of the closed-loop system
\[
\dot{x} = f(x, N(x), w) := f^c(x, w)
\]

\(u = N(x)\) is pre-trained feed-forward neural network with \(k\)-layer:
\[
\xi^{(i)}(x) = \phi^{(i)}(W^{(i-1)}\xi^{(i-1)}(x) + b^{(i-1)})
\]
\[
x = \xi^{(0)}, \quad u = W^{(k)}\xi^{(k)}(x) + b^{(k)} := N(x),
\]

**Rigorousness of control tools + effectiveness of ML tools**

Combine our reachability frameworks with neural network verification methods
**Input-output bounds:** Given a neural network controller \( u = N(x) \)

\[
\underline{u}[x,\bar{x}] \leq N(x) \leq \bar{u}[x,\bar{x}], \quad \text{for all } x \in [x, \bar{x}]
\]

\(^5\text{H. Zhang et al., NeurIPS 2018.}\)
Input-output bounds: Given a neural network controller $u = N(x)$

$$\bar{u}[x,\bar{x}] \leq N(x) \leq \bar{u}[x,\bar{x}], \text{ for all } x \in [x, \bar{x}]$$

Many neural network verification algorithms can produce these bounds. ex. CROWN (H. Zhang et al., 2018), LipSDP (M. Fazlyab et al., 2019), IBP (S. Gowal et al., 2018).
Input-output bounds: Given a neural network controller $u = N(x)$

$$u[l, u] \leq N(x) \leq \bar{u}[l, u], \text{ for all } x \in [l, u]$$

Many neural network verification algorithms can produce these bounds. ex. CROWN (H. Zhang et al., 2018), LipSDP (M. Fazlyab et al., 2019), IBP (S. Gowal et al., 2018).

CROWN\(^5\)

- Bounding the value of each neurons
- Linear upper and lower bounds on the activation function

Reachability of open-loop system treating $u$ as a parameter

Neural network verification algorithm for bounds on $u$

Reachability of open-loop system + Neural network verification bounds
Safety of Neural Network Controlled Systems
A Compositional Approach

\[
\dot{x} = d(x, \bar{x}, u, \bar{u}, w, \bar{w}) \\
\dot{x} = \bar{d}(x, \bar{x}, u, \bar{u}, w, \bar{w})
\]

\[
u_{[x, \bar{x}]} \leq N(x) \leq \bar{u}_{[x, \bar{x}]} \quad \text{for every } x \in [x, \bar{x}].
\]

\[
\dot{x} = d(x, \bar{x}, u_{[x, \bar{x]}}, \bar{u}_{[x, \bar{x]}}, w, \bar{w}) \\
\dot{x} = \bar{d}(x, \bar{x}, u_{[x, \bar{x]}}, \bar{u}_{[x, \bar{x]}}, w, \bar{w})
\]
\[
\dot{x} = d(x, \bar{x}, u, \bar{u}, w, \bar{w}) \\
\dot{\bar{x}} = \bar{d}(x, \bar{x}, u, \bar{u}, w, \bar{w})
\]

\[
u_{[x, \bar{x}]} \leq N(x) \leq \bar{u}_{[x, \bar{x}]} \quad \text{for every } x \in [x, \bar{x}].
\]

\[
\dot{x} = d(x, \bar{x}, u_{[x, \bar{x}]}, \bar{u}_{[x, \bar{x}]}, w, \bar{w}) \\
\dot{\bar{x}} = \bar{d}(x, \bar{x}, u_{[x, \bar{x}]}, \bar{u}_{[x, \bar{x}]}, w, \bar{w})
\]

**Composition approach over-approximation:**
\[
\mathcal{R}_{fe}(t, \mathcal{X}_0, \mathcal{W}) \subseteq [x(t), \bar{x}(t)]
\]
\[
\dot{x} = d(x, \bar{x}, u, \bar{u}, w, \bar{w})
\]

\[
\dot{x} = \overline{d}(x, \bar{x}, u, \bar{u}, w, \bar{w})
\]

\[
\underline{u}[x, \bar{x}] \leq N(x) \leq \overline{u}[x, \bar{x}] \quad \text{for every } x \in [x, \bar{x}].
\]

\[
\dot{x} = d(x, \bar{x}, \underline{u}[x, \bar{x}], \overline{u}[x, \bar{x}], w, \bar{w})
\]

\[
\dot{x} = \overline{d}(x, \bar{x}, \underline{u}[x, \bar{x}], \overline{u}[x, \bar{x}], w, \bar{w})
\]

Composition approach over-approximation:

\[
R_{fc}(t, \mathcal{X}_0, \mathcal{W}) \subseteq [x(t), \bar{x}(t)]
\]

It lead to overly-conservative estimates of reachable set
Dynamics of bicycle

\[
\begin{align*}
\dot{p}_x &= v \cos(\phi + \beta(u_2)) \\
\dot{\phi} &= \frac{v}{l_r} \sin(\beta(u_2)) \\
\dot{p}_y &= v \sin(\phi + \beta(u_2)) \\
\dot{v} &= u_1 \\
\beta(u_2) &= \arctan\left(\frac{l_r}{l_f + l_r} \tan(u_2)\right)
\end{align*}
\]
Dynamics of bicycle

\[
\begin{align*}
\dot{p}_x &= v \cos(\phi + \beta(u_2)) \\
\dot{p}_y &= v \sin(\phi + \beta(u_2)) \\
\dot{\phi} &= \frac{v}{l_r} \sin(\beta(u_2)) \\
\dot{v} &= u_1 \\
\beta(u_2) &= \arctan\left(\frac{l_r}{l_f + l_r} \tan(u_2)\right)
\end{align*}
\]

Goal: steer the bicycle to the origin avoiding the obstacles
Case Study: Bicycle Model
A naive compositional approach

Dynamics of bicycle

\[
p_x = v \cos(\phi + \beta(u_2)) \quad \dot{\phi} = \frac{v}{\ell_r} \sin(\beta(u_2))
\]

\[
p_y = v \sin(\phi + \beta(u_2)) \quad \dot{v} = u_1
\]

\[
\beta(u_2) = \arctan\left(\frac{l_r}{l_f + l_r} \tan(u_2)\right)
\]

**Goal:** steer the bicycle to the origin avoiding the obstacles

- train a feedforward neural network $4 \mapsto 100 \mapsto 100 \mapsto 2$ using data from model predictive control
Reachability of Closed-loop System
Case Study: Bicycle Model

- start from \((8, 8)\) toward \((0, 0)\)
- \(X_0 = [x_0, \bar{x}_0]\) with
  \[
  x_0 = \begin{pmatrix} 7.95 & 7.95 & -\frac{\pi}{3} - 0.01 & 1.99 \end{pmatrix}^T
  \]
  \[
  \bar{x}_0 = \begin{pmatrix} 8.05 & 8.05 & -\frac{\pi}{3} + 0.01 & 2.01 \end{pmatrix}^T
  \]
- CROWN for verification of neural network

Embedding system:

\[
\begin{align*}
\dot{x} &= d(x, \bar{x}, \bar{u}, \bar{u}, \bar{w}) \\
\dot{\bar{x}} &= \bar{d}(x, \bar{x}, \bar{u}, \bar{u}, \bar{w}, \bar{w})
\end{align*}
\]

\(u \leq N(x) \leq \bar{u}\), for every \(x \in [x, \bar{x}]\).
Reachability of Closed-loop System
Case Study: Bicycle Model

- start from \((8, 8)\) toward \((0, 0)\)
- \(X_0 = [x_0, \bar{x}_0]\) with
  \[
  x_0 = \begin{pmatrix} 7.95 & 7.95 & -\frac{\pi}{3} - 0.01 & 1.99 \end{pmatrix}^T
  \]
  \[
  \bar{x}_0 = \begin{pmatrix} 8.05 & 8.05 & -\frac{\pi}{3} + 0.01 & 2.01 \end{pmatrix}^T
  \]
- CROWN for verification of neural network

Euler integration with step \(h\):
\[
x_1 = x_0 + hd(x_0, \bar{x}_0, u_0, \bar{u}_0, w, \bar{w})
\]
\[
\bar{x}_1 = \bar{x}_0 + h\bar{d}(x_0, \bar{x}_0, u_0, \bar{u}_0, w, \bar{w})
\]
\[u_0 \leq N(x) \leq \bar{u}_0, \text{ for every } x \in [x_0, \bar{x}_0].\]
Reachability of Closed-loop System
Case Study: Bicycle Model

- Start from $(8, 8)$ toward $(0, 0)$
- $X_0 = [x_0, \bar{x}_0]$ with
  \[
  x_0 = (7.95 \quad 7.95 \quad -\frac{\pi}{3} - 0.01 \quad 1.99)^{\top}
  \]
  \[
  \bar{x}_0 = (8.05 \quad 8.05 \quad -\frac{\pi}{3} + 0.01 \quad 2.01)^{\top}
  \]
- CROWN for verification of neural network

Euler integration with step $h$:
\[
x_2 = x_1 + hd(x_1, \bar{x}_1, u_1, \bar{u}_1, w, \bar{w}) \]
\[
\bar{x}_2 = \bar{x}_1 + h\bar{d}(x_1, \bar{x}_1, u_1, \bar{u}_1, w, \bar{w})
\]
\[
u_1 \leq N(x) \leq \bar{u}_1, \text{ for every } x \in [x_1, \bar{x}_1].\]
start from \((8, 8)\) toward \((0, 0)\)

\[ X_0 = [x_0, \overline{x}_0] \]

\[
\begin{align*}
x_0 &= \begin{pmatrix} 7.95 & 7.95 & -\frac{\pi}{3} & -0.01 & 1.99 \end{pmatrix}^T \\
\overline{x}_0 &= \begin{pmatrix} 8.05 & 8.05 & -\frac{\pi}{3} & +0.01 & 2.01 \end{pmatrix}^T
\end{align*}
\]

- CROWN for verification of neural network

Euler integration with step \(h\):

\[
\begin{align*}
x_3 &= x_2 + hd(x_2, \overline{x}_2, u_2, \overline{u}_2, w, \overline{w}) \\
\overline{x}_3 &= \overline{x}_2 + \overline{h}d(x_2, \overline{x}_2, u_2, \overline{u}_2, w, \overline{w})
\end{align*}
\]

\[ u_2 \leq N(x) \leq \overline{u}_2, \text{ for every } x \in [x_2, \overline{x}_2]. \]
Neural network controller as disturbances (worst-case scenario)
It does not capture the stabilizing effect of the neural network.
Neural network controller as disturbances (worst-case scenario)
It does not capture the stabilizing effect of the neural network.

An illustrative example
\[ \dot{x} = x + u + w \] with controller \( u = -K x \), for some unknown \( 1 < K \leq 3 \).
Stabilizing Effect of Neural Network Controllers

Issues with the compositional approach

Neural network controller as **disturbances** (worst-case scenario)
It does not capture the **stabilizing** effect of the neural network.

**An illustrative example**
\[ \dot{x} = x + u + w \] with controller \( u = -Kx \), for some unknown \( 1 < K \leq 3 \).

<table>
<thead>
<tr>
<th>Compositional approach</th>
<th>Interaction-aware approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>First find the bounds ( \underline{u} \leq Kx \leq \overline{u} ), then</td>
<td></td>
</tr>
<tr>
<td>[ \dot{x} = x + \underline{u} + w ]</td>
<td></td>
</tr>
<tr>
<td>[ \dot{x} = \overline{x} + \underline{u} + \overline{w} ]</td>
<td></td>
</tr>
<tr>
<td>This system is unstable.</td>
<td></td>
</tr>
<tr>
<td>First replace ( u = Kx ) in the system, then</td>
<td></td>
</tr>
<tr>
<td>[ \dot{x} = (1 - K)x + w ]</td>
<td></td>
</tr>
<tr>
<td>[ \dot{x} = (1 - K)\overline{x} + \overline{w} ]</td>
<td></td>
</tr>
<tr>
<td>This system is stable.</td>
<td></td>
</tr>
</tbody>
</table>
Neural network controller as **disturbances** (worst-case scenario)
It does not capture the **stabilizing** effect of the neural network.

**An illustrative example**

\[ \dot{x} = x + u + w \]
with controller \( u = -Kx \), for some unknown \( 1 < K \leq 3 \).

**Compositional approach**

First find the bounds \( u \leq Kx \leq \bar{u} \), then

\[ \dot{x} = x + u + w \]
\[ \ddot{x} = \bar{x} + \bar{u} + \bar{w} \]

This system is unstable.

**Interaction-aware approach**

First replace \( u = Kx \) in the system, then

\[ \dot{x} = (1 - K)x + w \]
\[ \ddot{x} = (1 - K)x + \bar{w} \]

This system is stable.

We need to know the **functional** dependencies of neural network bounds.
**Functional bounds:** Given a neural network controller $u = N(x)$

$$N_{[x,x]}(x) \leq N(x) \leq N_{[x,x]}(x), \quad \text{for all } x \in [x, \bar{x}]$$

---

**Functional bounds:** Given a neural network controller \( u = N(x) \)

\[
N_{[x,\bar{x}]}(x) \leq N(x) \leq \overline{N}_{[x,\bar{x}]}(x), \quad \text{for all } x \in [x, \bar{x}]
\]

- **Example:** CROWN\(^6\) can provide functional bounds.

**CROWN functional bounds:**

\[
N_{[x,\bar{x}]}(x) = A_{[x,\bar{x}]}x + b_{[x,\bar{x}]},
\]

\[
\overline{N}_{[x,\bar{x}]}(x) = \overline{A}_{[x,\bar{x}]}x + \overline{b}_{[x,\bar{x}]}
\]

**CROWN input-output bounds:**

\[
u_{[x,\bar{x}]} = A_{[x,\bar{x}]}^+ \bar{x} + \overline{A}_{[x,\bar{x}]}^- x + b_{[x,\bar{x}]},
\]

\[
\bar{u}_{[x,\bar{x}]} = \overline{A}_{[x,\bar{x}]}^+ \bar{x} + \overline{A}_{[x,\bar{x}]}^- x + \overline{b}_{[x,\bar{x}]}
\]

---

\(^6\)H. Zhang et al., NeurIPS 2018.
Safety of Neural Network Controlled Systems

Interaction-aware Approach

Theorem

Original system
\[ \dot{x} = f(x, N(x), w) \]

Embedding system
\[
\begin{bmatrix}
\dot{x} \\
\dot{w}
\end{bmatrix} = \begin{bmatrix}
H^+ - J_{[x, x]}^- H^- & H^- - J_{[w, w]}^- 
\end{bmatrix} \begin{bmatrix}
x \\
w
\end{bmatrix} + \begin{bmatrix}
-J_{[x, w]}^- & J_{[x, w]}^+ \\
-J_{[w, w]}^- & J_{[w, w]}^+
\end{bmatrix} \begin{bmatrix}
w \\
w
\end{bmatrix} + Q
\]

\( H \) and \( \overline{H} \) capture the effect of interactions between nonlinear system and neural network.

Interaction-aware over-approximation:
\[ \mathcal{R}_{fe}(t, x_0, W) \subseteq [x(t), \overline{x}(t)] \]

\(^7\text{SJ} \) and A. Harapanahalli and S. Coogan, under review, 2023
• start from (8, 7) toward (0, 0)
• \( X_0 = [x_0, \bar{x}_0] \) with
  \[
  x_0 = \begin{pmatrix} 7.95 & 6.95 & -\frac{2\pi}{3} & -0.01 & 1.99 \end{pmatrix}^T
  \]
  \[
  \bar{x}_0 = \begin{pmatrix} 8.05 & 7.05 & -\frac{2\pi}{3} + 0.01 & 2.01 \end{pmatrix}^T
  \]
• CROWN for verification of neural network
Dynamics of the $j$th vehicle

\[
\begin{align*}
\dot{p}_{x}^{j} &= v_{x}^{j}, & \dot{v}_{x}^{j} &= \tanh(u_{x}^{j}) + w_{x}^{j}, \\
\dot{p}_{y}^{j} &= v_{y}^{j}, & \dot{v}_{y}^{j} &= \tanh(u_{y}^{j}) + w_{y}^{j},
\end{align*}
\]

where $w_{x}^{j}, w_{y}^{j} \sim \mathcal{U}([-0.001, 0.001])$. 
Dynamics of the \( j \)th vehicle

\[
\begin{align*}
\dot{p}_x^j &= v_x^j, \\
\dot{v}_x^j &= \tanh(u_x^j) + w_x^j, \\
\dot{p}_y^j &= v_y^j, \\
\dot{v}_y^j &= \tanh(u_y^j) + w_y^j,
\end{align*}
\]

where \( w_x^j, w_y^j \sim U([-0.001, 0.001]) \). First vehicle uses a neural network controller

\[
4 \times 100 \times 100 \times 2, \text{ with ReLU activations}
\]

and is trained using trajectory data from an MPC controller for the first vehicle.
Dynamics of the $j$th vehicle

\[
\begin{align*}
\dot{p}_x^j &= v_x^j, \\
\dot{v}_x^j &= \tanh(u_x^j) + w_x^j, \\
\dot{p}_y^j &= v_y^j, \\
\dot{v}_y^j &= \tanh(u_y^j) + w_y^j,
\end{align*}
\]

where $w_x^j, w_y^j \sim \mathcal{U}([-0.001, 0.001])$. Other vehicles use PD controller

\[
\begin{align*}
u_d^j &= k_p \left( p_d^{j-1} - p_d^j - r \frac{v_d^{j-1}}{\|v_d^{j-1}\|_2} \right) \\
&\quad + k_v(v_d^{j-1} - v_d^j),
\end{align*}
\]

where $d \in \{x, y\}$. 
Dynamics of the $j$th vehicle

\[
\begin{align*}
\dot{p}_x^j &= v_x^j, & \dot{v}_x^j &= \tanh(u_x^j) + w_x^j, \\
\dot{p}_y^j &= v_y^j, & \dot{v}_y^j &= \tanh(u_y^j) + w_y^j,
\end{align*}
\]

where $w_x^j, w_y^j \sim \mathcal{U}([-0.001, 0.001])$.

- compositional approach
- a platoon of 9 vehicles
- reachable overapproximations for $t \in [0, 1.5]$
Dynamics of the $j$th vehicle

$$\dot{p}_x^j = v_x^j, \quad \dot{v}_x^j = \tanh(w_x^j) + w_x^j,$$
$$\dot{p}_y^j = v_y^j, \quad \dot{v}_y^j = \tanh(w_y^j) + w_y^j,$$

where $w_x^j, w_y^j \sim \mathcal{U}([-0.001, 0.001])$.

- interaction-aware approach
- a platoon of 9 vehicles
- reachable over-approximations for $t \in [0, 1.5]$
Conclusions

Key takeaways

- reachability as a framework for safety certification of autonomous systems
- developed computationally efficient and scalable approaches for reachability using monotone system theory
- run-time verification of neural network controlled systems
- capture stabilizing effect of learning-based components