A CEGAR Approach for Stability Verification of Linear Hybrid Systems

Miriam García Soto
Co-authored work with Pavithra Prabhakar

DARS 2017
Cyber-Physical Systems (CPSs)

Systems in which software "cyber" interacts with the "physical" world

Medical Devices  Automotive  Robotics  Aeronautics  Process control

Software controlled physical systems

* Automotive systems: Cruise control, lane assistants
* Medical Devices: Pacemakers, infusion pumps

Critical aspects in CPS design

* Security
* Reliability
* Safety

Grand Challenge
How do we build and deploy robust CPS?
Formal Verification

- Models for Cyber-Physical Systems (Automata based)
- Robustness Specifications (Logic based)
- Verification Algorithms (Model checker)

![Diagram showing the process of formal verification involving models, specifications, verification algorithms, and outcomes (certificate or counterexample).]
CPS Model
Hybrid Systems capture one of the main features of CPS, the mixed **continuous** and **discrete** behaviour.
Cruise control & automatic gearbox

**Gearbox**

\[ \dot{v} = \frac{p_r^q T}{M} \]

**Discrete Variable**

Gear Position \( q \)

\( q = 1, 2, 3, 4 \)

**Continuous Variables**

Error \( E = (v_d - v) \)

Torque \( T \)

**Continuous Dynamics**

\[ \dot{E} = \frac{-p_r^q}{M} T \]

\[ \dot{T} = \frac{K_q}{T} E + K_q E \]

**Cruise control**

**PI Control**

\[ \frac{K_q}{\tau} \int (v_d - v) dv \]

\[ K_q (v_d - v) \]

**Discrete Control**

If \( v - v_d = \frac{1}{p_t} \omega_{low} \)

\( q \rightarrow q - 1 \)

If \( v - v_d = \frac{1}{p_t} \omega_{high} \)

\( q \rightarrow q + 1 \)

\( q \)
Hybrid Automata

\[
\begin{align*}
E &= \frac{1}{p_1} \omega_{\text{high}} \\
E &= \frac{1}{p_2} \omega_{\text{high}} \\
E &= \frac{1}{p_3} \omega_{\text{high}} \\
E &= \frac{1}{p_4} \omega_{\text{low}} \\
E &= \frac{1}{p_2} \omega_{\text{low}} \\
E &= \frac{1}{p_3} \omega_{\text{low}} \\
E &= \frac{1}{p_4} \omega_{\text{low}} \\
\dot{x} &= A_1 x \\
\dot{x} &= A_2 x \\
\dot{x} &= A_3 x \\
\dot{x} &= A_4 x
\end{align*}
\]

Trajectories

Executions
CPS Specifications
Specifications

**Stability**: Small perturbations in the initial state or input to the system result in only small deviations from the nominal behavior.

- Cruise control: stability with respect to the desired velocity
- Robotic arm: stability with respect to the set point
- Bipedal walking: stability with respect to the periodic orbit
A system is **Lyapunov stable** with respect to the **equilibrium point 0** if for every $\varepsilon > 0$ there exists $\delta > 0$ such that for every execution $\sigma$ starting from $B_\delta(0)$, $\sigma(t) \in B_\varepsilon(0)$, for all time $t$.

A system is **asymptotically stable** with respect to the **equilibrium point 0** if it is Lyapunov stable and there exist $\eta > 0$ such that every execution $\sigma$ starting from $B_\eta(0)$ converges to 0.

**Lyapunov Stable**  
**Asymptotically Stable**  
**Unstable**
Stability analysis challenges

**Linear dynamical systems**

Stability can be determined by eigenvalues analysis.

**Linear hybrid systems**

Eigenvalue analysis does not suffice for switched linear system.
State of the art: Lyapunov’s second method

Continuous dynamics:
\[ \dot{x} = F(x) \]

If there exists a Lyapunov function for the system, then the system is Lyapunov stable.

Lyapunov function

- Continuously differentiable
  \[ V : \mathbb{R}^n \rightarrow \mathbb{R}^+ \]
- Positive definite
  \[ V(x) \geq 0 \ \forall x \]
  \[ V(x) = 0 \text{ iff } x = 0 \]
- Function value decreases along any trajectory
  \[ \frac{\partial V(x)}{\partial x} F(x) \leq 0 \ \forall x \]

Switched and hybrid systems:

- Common Lyapunov functions
- Multiple Lyapunov functions
Automated analysis

**Template based automated search**

- Choose a template
- Encode Lyapunov function conditions as constraints
- Solve using sum-of-squares programming tools

**Shortcomings:**

- Success depends crucially on the choice of the template
- The current methods provide no insight into the reason for the failure, when a template fails to prove stability
- No guidance regarding the choice of the next template

Alternate approach

CEGAR
Counterexample Guided Abstraction Refinement (CEGAR)
CEGAR for stability

First CEGAR approach for stability verification of hybrid systems

**CEGAR framework**

- Systematically iterates over the abstract systems
- Returns a counterexample in the case that the abstraction fails
- The counterexample can be used to guide the choice of the next abstraction

**Template based search**

- Success depends crucially on the choice of the template
- The current methods provide no insight into the reason for the failure, when a template fails to prove stability
- No guidance regarding the choice of the next template
Quantitative Predicate Abstraction
Quantitative Predicate Abstraction

Facets $\mathcal{F} = \{f_1, f_2, f_3, f_4\}$

Concrete system
Concrete system

Facets $\mathcal{F} = \{f_1, f_2, f_3, f_4\}$
Quantitative Predicate Abstraction

Facets $\mathcal{F} = \{f_1, f_2, f_3, f_4\}$

Concrete system

Abstract system
Quantitative Predicate Abstraction

Concrete system
Facets $\mathcal{F} = \{f_1, f_2, f_3, f_4\}$

Abstract system
Quantitative Predicate Abstraction

Concrete system
Facets \( \mathcal{F} = \{ f_1, f_2, f_3, f_4 \} \)

Abstract system

An edge between facets indicates the existence of an execution.
An edge between facets indicates the existence of an execution.
Quantitative Predicate Abstraction

Facets $\mathcal{F} = \{f_1, f_2, f_3, f_4\}$

Concrete system

Abstract system

An edge between facets indicates the existence of an execution.

Weights capture information about distance to the equilibrium point along the executions.
Quantitative Predicate Abstraction

Facets $\mathcal{F} = \{f_1, f_2, f_3, f_4\}$

Concrete system

Abstract system

An edge between facets indicates the existence of an execution.

Weights capture information about distance to the equilibrium point along the executions.
Quantitative Predicate Abstraction

A graph representation with nodes and edges.

Concrete system
Facets $\mathcal{F} = \{f_1, f_2, f_3, f_4\}$

Abstract system
Weights $W(\pi) = 2 \cdot \frac{1}{3} \cdot \frac{1}{3} \cdot 1 = \frac{2}{9} < 1$

An edge between facets indicates the existence of an execution.

Weights capture information about distance to the equilibrium point along the executions.
Quantitative Predicate Abstraction - samples

Product of edge weights = 1
Lyapunov Stable

Product of edge weights = 1/4
Asymptotically Stable

Product of edge weights = 4
Unstable
Weight computation

Constant dynamics \( \dot{x} = c \)

\[ f_2 \]
\[ f_1 \]
\[ \alpha d_2 \]
\[ \alpha d_1 \]
\[ d_2 \]
\[ d_1 \]

\[ \frac{|d_2|}{|d_1|} = \frac{|\alpha d_2|}{|\alpha d_1|} \]

Higher dimensions

\[ \frac{|\vec{b}|}{|\vec{a}|} \neq \frac{|\vec{b} + \vec{d}|}{|\vec{a} + \vec{d}|} \]

Weight (LP problems)

\[ \sup \frac{|v_2|}{|v_1|} \]
\[ t \geq 0, v_1 \in f_1, v_2 \in f_2, v_2 = v_1 + ct \]
Polyhedral inclusion dynamics $\dot{x} \in P$

$P$ is a polyhedral set

Weight (LP problems)

$$\sup \frac{|v_2|}{|v_1|} \quad \land a_i \cdot (v_2 - v_1) \leq b_i t$$

$t \geq 0, v_1 \in f_1, v_2 \in f_2, v_2 - v_1 + ct, \land a_i \cdot c \leq b_i$
Weight computation

**Linear dynamics** $\dot{x} = Ax$

<table>
<thead>
<tr>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\sup \frac{</td>
</tr>
</tbody>
</table>

$t \geq 0, v_1 \in f_1, v_2 \in f_2, v_2 = v_1 e^{At}$

- Solution is an exponential function
- Need a representation on which optimization can be performed
- Approximation methods [Girard et al., Frehse et al.]
Hybridization
Hybridization and soundness

Linear hybrid system

\[ \dot{x} = Ax \]

Polyhedral hybrid system

\[ \dot{x} \in P \]

**Theorem - Hybridization**

If the hybridized polyhedral hybrid system is Lyapunov (asymptotically) stable then the original linear hybrid system is Lyapunov (asymptotically) stable.

Hybridization for stability analysis of switched linear systems. **HSCC’16**
Theorem - Model-checking

A polyhedral hybrid system is Lyapunov stable if

* the abstract weighted graph has no edges with infinite weights, and
* no cycles with product of edge weights greater than 1

**Every cycle has weight smaller than 1**

=> **Concrete system is stable** => Stop

**There is a cycle, \( \pi \), with weight greater than 1**

=> **\( \pi \) is an abstract counterexample**

=> Validation

Abstraction based model-checking of stability of hybrid systems. CAV’13

Foundations of Quantitative Predicate Abstraction for Stability Analysis of Hybrid Systems. VMCAI’15
Counterexample

- Model-checking of the abstract system returns an abstract counterexample if the abstract system fails to establish stability.

  **Abstract Counterexample (ACE):**
  A cycle with product of edge weights greater than 1

- **Spurious ACE:** If there exist no infinite execution (concrete) of the system which follows the edges and weights of the cycle (and diverges)

- **Validation:** Checking if the ACE is spurious.

  **Validation is not a bounded model-checking problem!**
  Requires checking for an infinite execution instead of a finite execution.
Validation
Existence of an infinite concrete counterexample is equivalent to the existence of a finite execution along the cycle with certain properties, which can be encoded as an SMT formula.
Refinement
Counterexample guided abstraction refinement for stability analysis. CAV’16
Software tool
AVERIST flowchart and software dependencies

Stability Verifier

HYBRIDIZATION → PHS

AVERIST

ABSTRACTION

MODEL-CHECKING

VALIDATION

REFINEMENT

Stable/Unstable/Abstract counterexample

PPL

GLPK

NetworkX

Z3

http://software.imdea.org/projects/averist/index.html
Conclusion
Development of a novel **CEGAR approach**, based on abstraction and model-checking techniques

- **Automatic** process for linear and **polyhedral hybrid systems**
- **Framework extendable** to more complex class of hybrid systems
- Techniques implemented in **AVERIST** provide promising results
- Application to an **automatic gearbox**
Stability Verifier

LHS → HYBRIDIZATION → PHS → ABSTRACTION → MODEL-CHECKING → VALIDATION → REFINEMENT → Stable/Unstable/Abstract counterexample

http://software.imdea.org/projects/averist/index.html