An Introduction to Hybrid Automata, Numerical Simulation and Reachability Analysis

Goran Frehse

SyDe Summer School, September 10, 2015

Univ. Grenoble Alpes – Verimag,
2 avenue de Vignate, Centre Equation,
38610 Gières, France,
frehse@imag.fr

Overview

Hybrid Automata

Numerical Simulation

Set-Based Reachability

Conclusions

Overview

Hybrid Automata

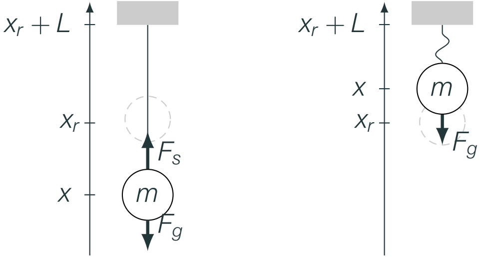
- Running Example
- Definition and Semantics

Numerical Simulation

Set-Based Reachability

Conclusions

Running Example: Ball on String



(a) *extension* (b) *freefall*

Equations of motion

- dynamics in *freefall* when $x \geq x_r$, with mass m,

$$m\ddot{x} = F_g = -mg.$$

- dynamics in *extension* when $x \leq x_r$, with spring constant k, damping factor d,

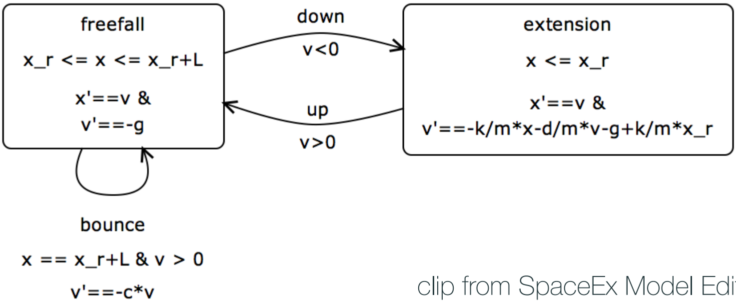
$$m\ddot{x} = F_g + F_s = -mg + kx_r - kx - d\dot{x}.$$

- transition when $x = x_r + L$, collision factor $c \in [0, 1]$,

$$\dot{x}' = -c\dot{x}.$$

Hybrid automaton model

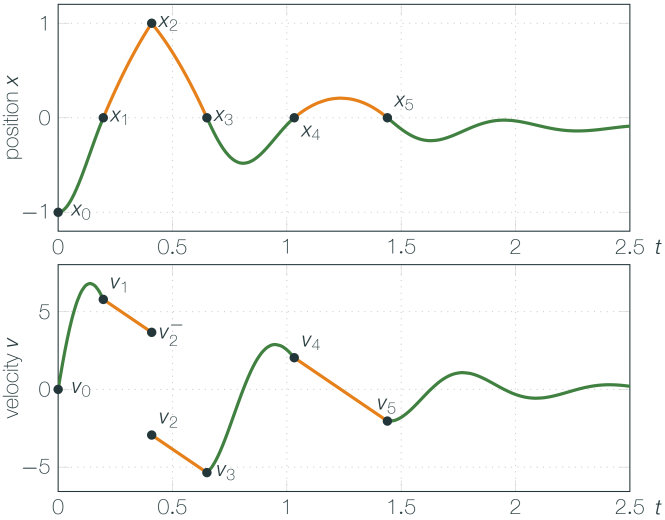
auxiliary variable $v = \dot{x}$, so $\dot{v} = \ddot{x}$.



clip from SpaceEx Model Editor[1]

[1] G. Frehse, C. L. Guernic, A. Donzé, R. Ray, O. Lebeltel, R. Ripado, A. Girard, T. Dang, and O. Maler, "Spaceex: Scalable verification of hybrid systems," in *CAV'11*, ser. LNCS, Springer, 2011.

Behavior



position x
1
0
−1
x_0
x_1
x_2
x_3
x_4
x_5
0
0.5
1
1.5
2
2.5
t
velocity v
5
0
−5
v_0
v_1
v_2^-
v_2
v_3
v_4
v_5
0
0.5
1
1.5
2
2.5
t

Overview

Hybrid Automata

- Running Example
- Definition and Semantics

Numerical Simulation

Set-Based Reachability

Conclusions

Hybrid Automata (Alur, Henzinger, '95)[2][3]

- locations $\mathrm{Loc} = \{\ell_1, \ldots, \ell_m\}$ and variables $X = \{x_1, \ldots, x_n\}$ define the state space $\mathrm{Loc} \times \mathbb{R}^X$,
- transitions $\mathrm{Edg} \subseteq \mathrm{Loc} \times \mathrm{Lab} \times \mathrm{Loc}$ define location changes with synchronization labels Lab,
- invariant or staying condition $\mathrm{Inv} \subseteq \mathrm{Loc} \times \mathbb{R}^X$,
- flow relation Flow, where $\mathrm{Flow}(\ell) \subseteq \mathbb{R}^{\dot{X}} \times \mathbb{R}^X$, e.g.,

$$\dot{\mathbf{x}} = f(\mathbf{x});$$

- jump relation Jump, where $\mathrm{Jump}(e) \subseteq \mathbb{R}^X \times \mathbb{R}^{X'}$, e.g.,

$$\mathrm{Jump}(e) = \{(\mathbf{x}, \mathbf{x}') \mid \mathbf{x} \in \mathcal{G} \wedge \mathbf{x}' = r(\mathbf{x})\},$$

- initial states $\mathrm{Init} \subseteq \mathrm{Inv}$.

Run Semantics

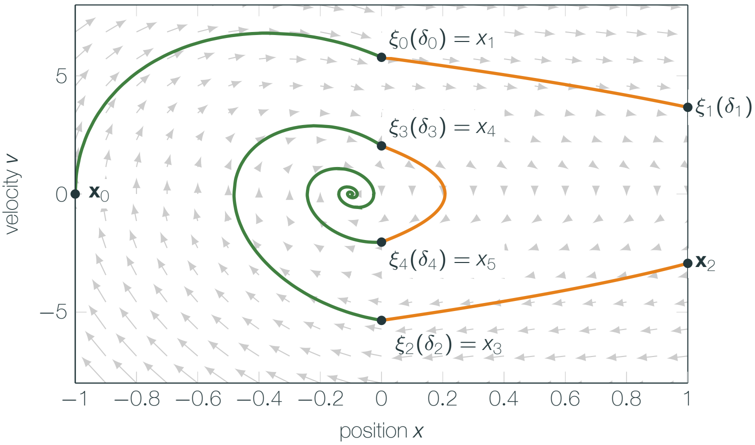
$$(\ell_0, \mathbf{x}_0) \xrightarrow{\delta_0, \xi_0} (\ell_0, \xi_0(\delta_0)) \xrightarrow{\alpha_0} (\ell_1, \mathbf{x}_1) \xrightarrow{\delta_1, \xi_1} (\ell_1, \xi_1(\delta_1)) \ldots$$

with $(\ell_0, \mathbf{x}_0) \in \mathrm{Init}$, $\alpha_i \in \mathrm{Lab} \cup \{\tau\}$, and for $i = 0, 1, \ldots$:

1. **Trajectories:** $(\dot{\xi}(t), \xi(t)) \in \mathrm{Flow}(\ell)$ and $\xi_i(t) \in \mathrm{Inv}(\ell_i)$ for all $t \in [0, \delta_i]$.
2. **Jumps:** $(\xi_i(\delta_i), \mathbf{x}_{i+1}) \in \mathrm{Jump}(e_i)$, $e_i = (\ell_i, \alpha_i, \ell_{i+1}) \in \mathrm{Edg}$, and $\mathbf{x}_{i+1} \in \mathrm{Inv}(\ell_{i+1})$.

A state $(\ell, \mathbf{x})$ is **reachable** if there exists a run with $(\ell_i, \mathbf{x}_i) = (\ell, \mathbf{x})$ for some i.

Example: Ball on String



Overview

Hybrid Automata

Numerical Simulation

- Solving ODEs
- Computing Trajectories and Jumps

Set-Based Reachability

Conclusions

Solving ODEs

Given an ordinary differential equation (ODE)

$$\dot{\mathbf{x}} = f(\mathbf{x}), \quad \text{with initial value } \mathbf{x}_0,$$

find $\xi(t)$ with $\xi(0) = \mathbf{x}_0$ and $\dot{\xi}(t) = f(\xi(t))$ for all $t \geq 0$.

Numerical solution by computing $\mathbf{x}_0, \ldots, \mathbf{x}_N$ such that $\mathbf{x}_i \approx \xi(t_i)$ at time points $t_0, \ldots, t_N$.[2]

Using fixed time step h: $t_i = ih$.

[2] R. P. Canale and S. C. Chapra, "Numerical methods for engineers," *Mc Graw Hill, New York*, 1998.

Euler's Method

Compute $\mathbf{x}_0, \ldots, \mathbf{x}_N$ with the sequence

$$x_{i+1} = x_i + f(x_i)h.$$

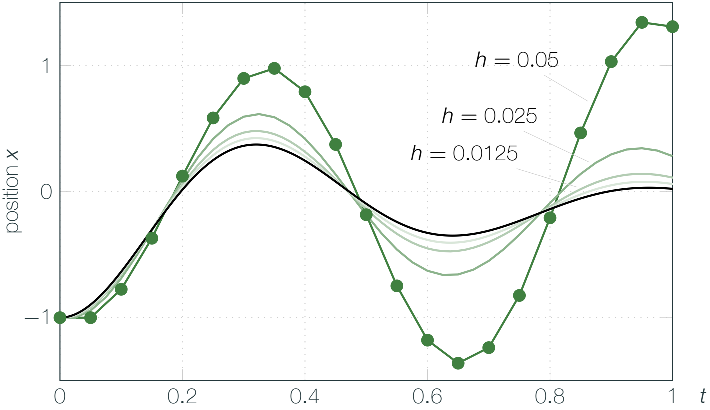
Comparing to Taylor series around x_i,

$$x_{i+1} = x_i + \dot{x}_i h + \frac{\ddot{x}_i}{2!}h^2 + \ldots + \frac{\overset{(n-1)}{x_i}}{n!}h^n + \cdots,$$

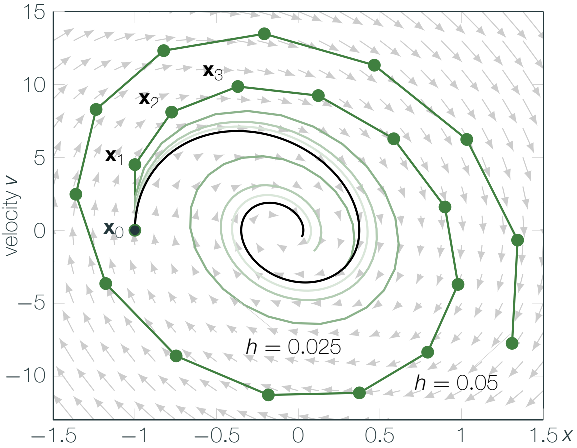
obtain estimate of local error $\varepsilon_a = \mathcal{O}(h^2)$.

- global error $\varepsilon_g = \mathcal{O}(h) \Rightarrow$ first-order method
- accuracy limited by numerical roundoff error $\mathcal{O}(1/h)$

Ball on String in Extension: Euler's Method



Ball on String in Extension: Euler's Method



Stability

The linear ODE

$$\dot{x} = ax,$$

converges to zero iff $a < 0$.

Euler's method

$$x_{i+1} = x_i + f(x_i)h = x_i + ax_ih = (1 + ah)x_i$$

converges to zero iff $|1 + ah| < 1 \Rightarrow$ **conditionally stable**.

Backwards Euler Method

Compute $\mathbf{x}_0, \ldots, \mathbf{x}_N$ with the sequence

$$x_{i+1} = x_i + f(x_{i+1})h,$$

solved for x_{i+1} at each i using root-finding (Newton's method).
$\Rightarrow$ implicit method

Backwards Euler for $\dot{x} = ax$,

$$x_{i+1} = x_i + ax_{i+1}h = \frac{1}{1 - ah}x_i$$

converges for all $a < 0$, $h > 0 \Rightarrow$ unconditionally stable.

Runge-Kutta Methods

Explicit Runge-Kutta methods compute the sequence

$$x_{i+1} = x_i + \phi(x_i, h)h,$$

$$\phi(x_i, h) = a_1 k_1 + a_2 k_2 + \cdots + a_n k_n,$$

weights a_i,q_{ij} and derivative $k_j = f(\hat{x}_i^j)$ at intermediate states

$$\begin{aligned}
\hat{x}_i^{\,1} &= x_i, \\
\hat{x}_i^{\,2} &= x_i + q_{11} k_1 h, \\
\hat{x}_i^{\,3} &= x_i + q_{21} k_1 h + q_{21} k_2 h, \\
&\vdots \\
\hat{x}_i^{\,n} &= x_i + q_{(n-1)1} k_1 h + q_{(n-1)2} k_2 h + \cdots + q_{(n-1)(n-1)} k_{n-1} h
\end{aligned}$$

Runge-Kutta Methods (Kutta, 1901)

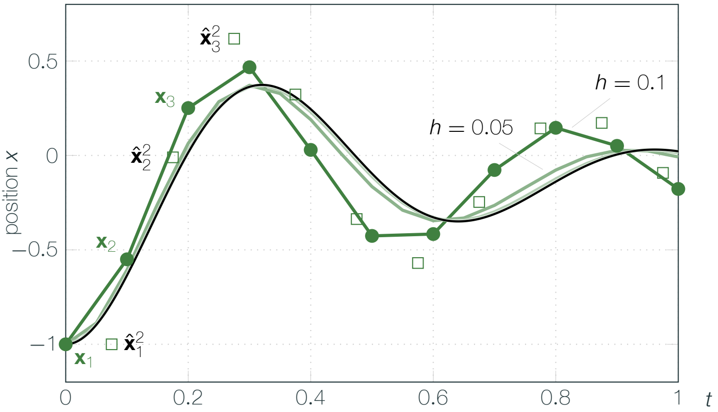
Runge-Kutta method defined by n and parameters a_i,q_{ij} chosen to match first n terms of Taylor series.

Remaining degrees of freedom used to optimize, e.g., truncation error $\mathcal{O}(h^{n+1})$ and global error $\mathcal{O}(h^n)$ for $n = 2, \ldots, 5$.

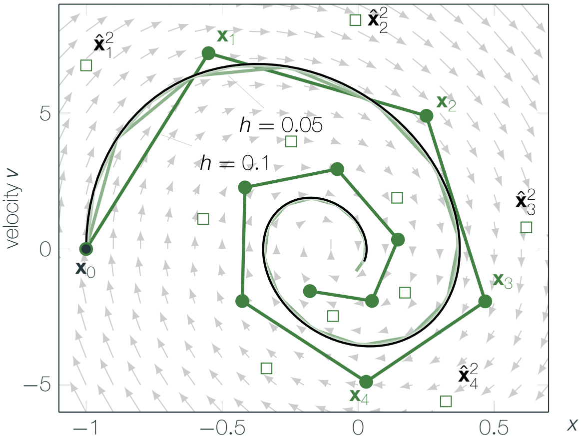
Ralston's method has the smallest truncation error for $n = 2$:

$$
\begin{aligned}
k_1 &= f(\hat{x}_i^1), & \hat{x}_i^1 &= x_i, \\
k_2 &= f(\hat{x}_i^2), & \hat{x}_i^2 &= x_i + 3/4 k_1 h, \\
\phi(x_i, h) &= \tfrac{1}{3}k_1 + \tfrac{2}{3}k_2.
\end{aligned}
$$

Ball on String in Extension: Ralston's Method



Ball on String in Extension: Ralston's Method



Error Estimation

Error estimation using difference between

- results for time steps $h/2$ and h,
- results for $(n-1)$th and nth order solver.

Runge-Kutta-Fehlberg (RKF) methods (Fehlberg, 1970)

- compare $(n-1)$th and nth order RK,
- reuse intermediate results,
- no more evaluations than nth order RK.

Popular: RKF 2(3) and RKF 4(5), aka `ode23` and `ode45`.

Adaptive Time Steps

Adapt time step using error estimation ε_a and desired error ϵ_d.

Heuristics[3]:

$$h \leftarrow h \left| \epsilon_d / \varepsilon_a \right|^{0.25} \text{ if } \varepsilon_a < \epsilon_d$$

$$h \leftarrow h \left| \epsilon_d / \varepsilon_a \right|^{0.2} \text{ if } \varepsilon_a \geq \epsilon_d$$

[3] W. H. Press, *Numerical recipes 3rd edition: The art of scientific computing*. Cambridge University Press, 2007.

Stiff Systems

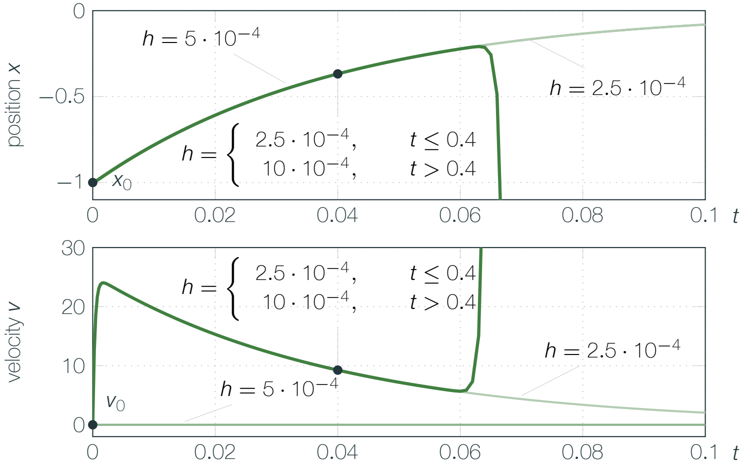
Stiff ODEs have time constants (Eigenvalues) differing by a factor of 1000 or more.

Solvers take tiny time steps throughout the entire time horizon

Special solvers are available for stiff ODEs, using implicit methods to achieve stability at larger time steps.

Ball on String example is stiff for small mass.

Ball on String: Stiff for Small Mass



Overview

Hybrid Automata

Numerical Simulation

- Solving ODEs
- Computing Trajectories and Jumps

Set-Based Reachability

Conclusions

Computing Trajectories and Jumps

Numerical simulation of hybrid automata:

- use ODE solver to approximate trajectories,
- detect when trajectory enters guard,
- detect when trajectory leaves invariant.

ODE solver offer **zero crossing detection** using root-finding algorithms.

Detecting guards/invariants using root functions is computationally expensive and potentially inaccurate.

Shortcomings[4]

- **Missed roots**
  violations of invariant or entering guard go undetected.
- **Increased cost**
  ODE solvers reuse intermediate states for increasing time sequence. Lost through back-and-forth of root-finding.
- **Spurious behavior**
  Numerically approximated state may lie slightly outside the guard or invariant, so constraints are relaxed

[4] F. Zhang, M. Yeddanapudi, and P. Mosterman, "Zero-crossing location and detection algorithms for hybrid system simulation," in *IFAC World Congress*, 2008, pp. 7967–7972.

Zeno Behavior

Zeno behavior occurs if infinitely many events occur in a bounded time interval.[5]

Chattering Zeno: zero-time events

Genuine Zeno: event times converge towards a fixed point

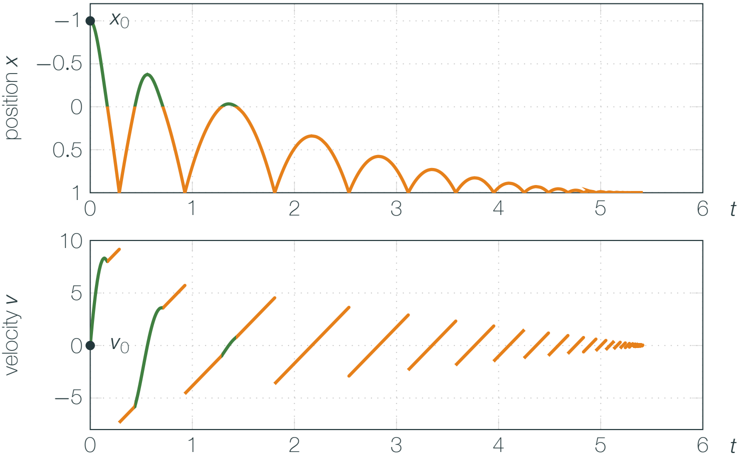
Simulator seems to get "stuck" as switching times converge.

Ball on String example zeno if upside-down (negative gravity) $\Rightarrow$ bouncing ball.

---

[5] A. D. Ames and S. Sastry, "Characterization of zeno behavior in hybrid systems using homological methods," in *ACC'05*, 2005.

Ball on String: Zeno Behavior



Accounting for Nondeterminism

The biggest challenge is nondeterminism:

- select initial state and successor states in jump relation;
- choose between transitions if guards overlap;
- choose jump time from interval of time;
- **differential inclusions**, such as $\dot{x} \in [-1, 1]$, require to pick derivative for **each time step**;

Number of runs increases exponentially with each choice.

Simulators like Simulink, Modelica, or Ptolemy use purely deterministic models that jump as soon as possible.

Overview

Hybrid Automata

Numerical Simulation

Set-Based Reachability

- Piecewise Constant Dynamics
- Piecewise Affine Dynamics
- Set Representations
- SpaceEx (advertisement)

Conclusions

Set-Based Reachability

Extending numerical simulation from numbers to sets

- account for nondeterminism
- exhaustive
- infinite time horizon

Downsides:

- only approximate for complex dynamics
- generally not scalable in # of variables
- trade-off between runtime and accuracy

Reachability Algorithm

One-step successors by time elapse from set of states S,

$$\mathrm{Post}_C(S) = \left\{ (\ell, \xi(\delta)) \;\middle|\; \exists(\ell, x) \in S : (\ell, \mathbf{x}) \xrightarrow{\delta, \xi} (\ell, \xi(\delta)) \right\}.$$

One-step successors by jump from set of states S,

$$\mathrm{Post}_D(S) = \Big\{ (\ell', \mathbf{x}') \;\Big|\; \exists(\ell', \mathbf{x}') \in S, \exists \alpha \in \mathrm{Lab} \cup \{\tau\} : \\ (\ell, \mathbf{x}) \overset{\alpha}{\to} (\ell', \mathbf{x}') \Big\}.$$

Reachability Algorithm

Compute sequence

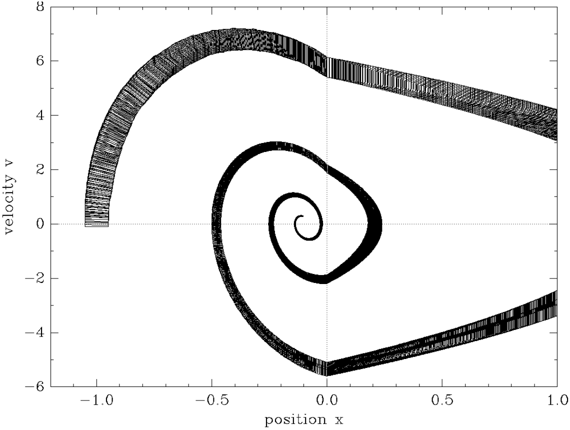
$$\begin{aligned} R_0 &= \text{Post}_C(\text{Init}), \\ R_{i+1} &= R_i \cup \text{Post}_C(\text{Post}_D(R_i)). \end{aligned}$$

If $R_{i+1} = R_i$, then R_i = reachable states.

- may not terminate if states unbounded (counter)
- problem undecidable in general[6]

[6] T. A. Henzinger, P. W. Kopke, A. Puri, and P. Varaiya, "What's decidable about hybrid automata?" *Journal of Computer and System Sciences*, vol. 57, pp. 94–124, 1998.

Ball on String: Reachable States



(clip from SpaceEx output)

HA with piecewise constant dynamics (PCDA)

- initial states and invariants given by conjunctions of linear constraints,
- flows given by conjunctions of linear constraints over the derivatives $\dot{X}$, and
- jumps given by linear constraints over $X \cup X'$, where X' denote the variables after the jump.

One-step successors of PCDA can be computed exactly.

Polyhedra in Constraint Form

$\mathcal{H}$-polyhedron (constraint form)

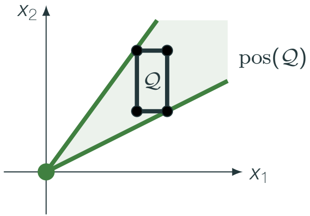
$$\mathcal{P} = \left\{ \mathbf{x} \;\middle|\; \bigwedge_{i=1}^{m} \mathbf{a}_i^\top \mathbf{x} \leq b_i \right\},$$

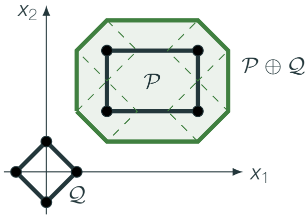
with facet normals $\mathbf{a}_i \in \mathbb{R}^n$ and inhomogeneous coefficients $b_i \in \mathbb{R}$.

vector-matrix notation:

$$\mathcal{P} = \left\{ \mathbf{x} \;\middle|\; A\mathbf{x} \leq \mathbf{b} \right\}, \text{ with } A = \begin{pmatrix} \mathbf{a}_1^\top \\ \vdots \\ \mathbf{a}_m^\top \end{pmatrix}, \mathbf{b} = \begin{pmatrix} b_1 \\ \vdots \\ b_m \end{pmatrix}.$$

Geometric Operations





The convex hull
$\mathrm{chull}(\mathcal{Q}) = \left\{ \sum_{\mathbf{q}_i \in \mathcal{Q}} \lambda_i \cdot \mathbf{q}_i \;\middle|\; \lambda_i \geq 0, \sum_i \lambda_i = 1 \right\}$,

The cone of $\mathcal{Q}$ is $\mathrm{pos}(\mathcal{Q}) = \{\mathbf{q} \cdot t \mid \mathbf{q} \in \mathcal{Q}, t \geq 0\}$.

The Minkowski sum is $\mathcal{P} \oplus \mathcal{Q} = \{\mathbf{p} + \mathbf{q} \mid \mathbf{p} \in \mathcal{P}, \mathbf{q} \in \mathcal{Q}\}$.

Polyhedra in Generator Form

$\mathcal{V}$-polyhedron (generator form)

$$\mathcal{P} = (V, R) = \mathrm{chull}\,(V) \oplus \mathrm{pos}(\mathrm{chull}(R)).$$

with vertices $V \subseteq \mathbb{R}^n$ and rays $R \subseteq \mathbb{R}^n$

conversion between $\mathcal{H}$- and $\mathcal{V}$-polyhedra is expensive

cube: $2n$ constraints, 2^n vertices

cross-polytope (diamond): $2n$ vertices, 2^n constraints

Time Elapse with Polyhedra

For PCDA, it suffices to consider straight-line trajectories:

Lemma (Constant Derivatives[7])

There is a trajectory $\xi(t)$ *from* $\mathbf{x} = \xi(0)$ *to* $\mathbf{x}' = \xi(\delta)$, $\delta > 0$, *iff* $\eta(t) = \mathbf{x} + \mathbf{q}t$ *with* $\mathbf{q} = (\mathbf{x}' - \mathbf{x})/\delta$ *is a trajectory from* $\mathbf{x}$ *to* $\mathbf{x}'$.

[7] P.-H. Ho, "Automatic analysis of hybrid systems," Technical Report CSD-TR95-1536, PhD thesis, Cornell University, Aug. 1995.

Time Elapse with Polyhedra

Given polyhedra $\mathcal{P} = \{\mathbf{x} \mid A\mathbf{x} \leq \mathbf{b}\}$, $\mathcal{Q} = \{\mathbf{q} \mid \bar{A}\mathbf{q} \leq \bar{\mathbf{b}}\}$

Time successors (without invariant):

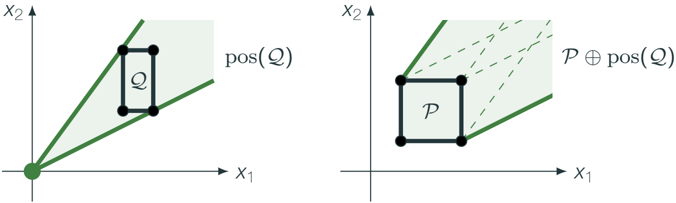
$$\mathcal{P} \nearrow \mathcal{Q} = \{\mathbf{x}' \mid \mathbf{x} \in \mathcal{P}, \mathbf{q} \in \mathcal{Q}, t \in \mathbb{R}^{\geq 0}, \mathbf{x}' = \mathbf{x} + \mathbf{q}t\}.$$

Eliminating $\mathbf{q} = \frac{\mathbf{x}' - \mathbf{x}}{t}$ for $t > 0$ and multiplying with t:

$$\mathcal{P} \nearrow \mathcal{Q} = \left\{\mathbf{x}' \;\middle|\; A\mathbf{x} \leq \mathbf{b} \;\wedge\; \bar{A}(\mathbf{x}' - \mathbf{x}) \leq \bar{\mathbf{b}} \cdot t \;\wedge\; t \geq 0\right\}.$$

Quantifier elimination of t squares the number of constraints.

Time Elapse with Polyhedra – Geometric Version



(c) cone $\mathrm{pos}(\mathcal{Q})$

(d) $\mathcal{P} \nearrow \mathcal{Q} = \mathcal{P} \oplus \mathrm{pos}(\mathcal{Q})$

Intersect with invariant:

$$\mathrm{post}_C(\ell \times P) = \ell \times \left(P \nearrow \mathrm{Flow}(\ell)\right) \cap \mathrm{Inv}(\ell).$$

Discrete Successors

Edge $e = (\ell, \alpha, \ell')$ with guard $\mathbf{x} \in \mathcal{G}$ and nondeterministic assignment $\mathbf{x}' = C\mathbf{x} + \mathbf{w}$, $\mathbf{w} \in \mathcal{W}$,

$$\mathrm{post}_D(\ell \times P) = \ell' \times \big(C(\mathcal{P} \cap \mathcal{G}) \oplus \mathcal{W}\big) \cap \mathrm{Inv}(\ell').$$

If linear map C singular, constraints require quantifier elimination, otherwise

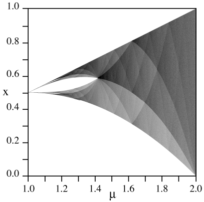
$$C\mathcal{P} = \{\mathbf{x} \mid AC^{-1}\mathbf{x} \leq b\}$$

Computational Cost

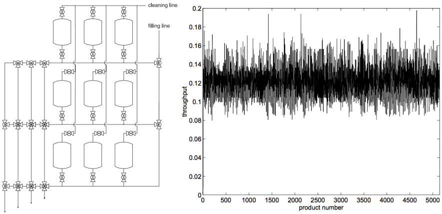
operation	polyhedra m constraints	 k generators
cone	m^2	k
Minkowski sum	exp	k^2
linear map	m / exp	k
intersection	$2m$	exp

Complex Behavior in PCDA

- **chaos**
  - even with 1 variable, 1 location, 1 transition (tent map)
  - observed in actual production systems [Schmitz,2002]

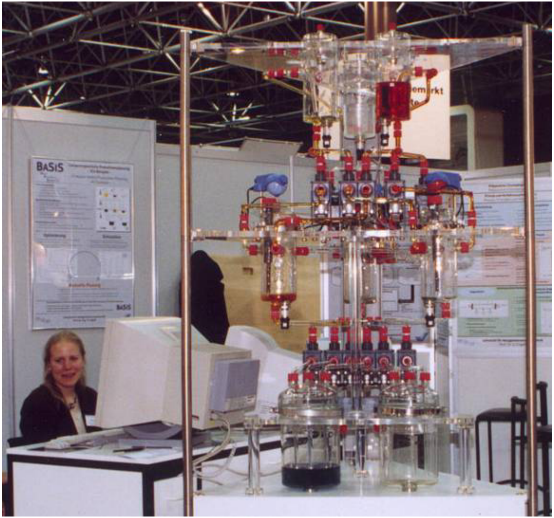


states of the Tent map
source: wikipedia

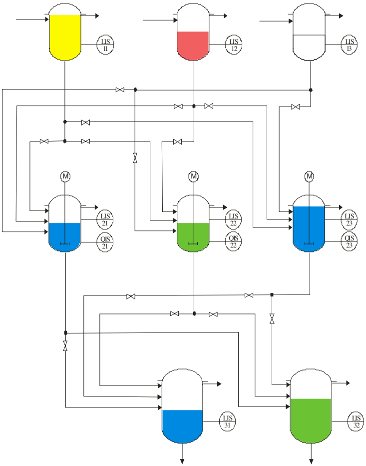


brewery and chaotic throughput [Schmitz,2002]

Example: Multi-Product Batch Plant

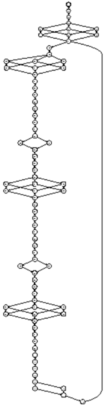


Example: Multi-Product Batch Plant

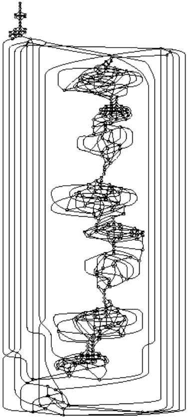


- **Cascade mixing process**
  - 3 educts via 3 reactors
    $\Rightarrow$ 2 products
- **Verification Goals**
  - Invariants
    - overflow
    - product tanks never empty
  - Filling sequence
- **Design of verified controller**

Verification with PHAVer



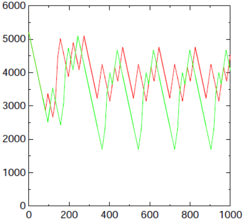
Controller



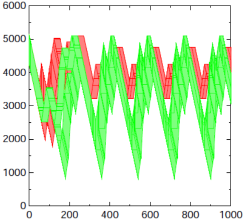
Controlled Plant

- **Controller + Plant**
  - 266 locations, 823 transitions (~150 reachable)
  - 8 continuous variables
- **Reachability over infinite time**
  - 120s—1243s, 260—600MB
  - computation cost increases with nondeterminism (intervals for throughputs, initial states)

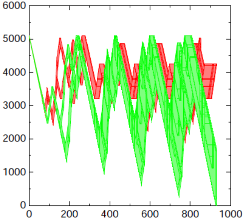
Verification with PHAVer



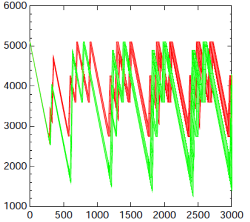
(a) BP8.1: nominal case



(b) BP8.2: varying initial cond.



(c) BP8.3: varying demand



(d) BP8.4: varying but slow demand

					Automaton		Reachable Set	
Instance	Time [s]	Mem. [MB]	Depth[a]	Checks[b]	Loc.	Trans.	Loc.	Poly.
BP8.1	120	267	173	279	266	823	130	279
BP8.2	139	267	173	422	266	823	131	450
BP8.3	845	622	302	2669	266	823	143	2737
BP8.4	1243	622	1071	4727	266	823	147	4772

* on Xeon 3.20 GHz, 4GB RAM running Linux; [a] lower bound on depth in breadth-first search; [b] number of applications of post-operator

Overview

- Hybrid Automata
- Numerical Simulation
- Set-Based Reachability
  - Piecewise Constant Dynamics
  - Piecewise Affine Dynamics
  - Set Representations
  - SpaceEx (advertisement)
- Conclusions

Piecewise Affine Dynamics

Hybrid automata with piecewise affine dynamics (PWA)

- initial states and invariants are polyhedra,
- flows are affine ODEs

$$\dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u}, \qquad \mathbf{u} \in \mathcal{U},$$

- jumps have a guard set and assignments

$$\mathbf{x}' = C\mathbf{x} + \mathbf{w}, \qquad \mathbf{w} \in \mathcal{W}.$$

Continuous successors

$$\dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u}, \qquad \mathbf{u} \in \mathcal{U},$$

trajectory $\xi(t)$ from $\xi(0) = \mathbf{x}_0$ for given input signal $\zeta(t) \in \mathcal{U}$:

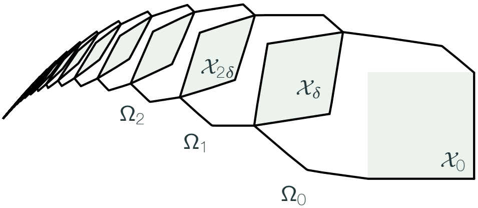
$$\xi_{\mathbf{x}_0,\zeta}(t) = e^{At}\mathbf{x}_0 + \int_0^t e^{A(t-s)}B\zeta(s)ds.$$

reachable states from set $\mathcal{X}_0$ for any input signal:

$$\mathcal{X}_t = e^{At}\mathcal{X}_0 \oplus \mathcal{Y}_t,$$

$$\mathcal{Y}_t = \int_0^t e^{As}\mathcal{U}ds = e^{At}\mathcal{X}_0 \oplus \lim_{\delta \to 0} \bigoplus_{k=0}^{\lfloor t/\delta \rfloor} e^{A\delta k}\delta\mathcal{U}.$$

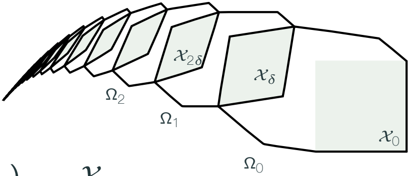
Computing a Convex Cover



Compute $\Omega_0, \Omega_1, \ldots$ such that

$$\bigcup_{0 \leq t \leq T} \mathcal{X}_t \subseteq \Omega_0 \cup \Omega_1 \cup \ldots.$$

Time Discretization



Semi-group property: $(\mathcal{X}_{k\delta})_\delta = \mathcal{X}_{(k+1)\delta}$

Time discretization: $\mathcal{X}_{(k+1)\delta} = e^{A\delta}\mathcal{X}_{k\delta} \oplus \mathcal{Y}_\delta.$

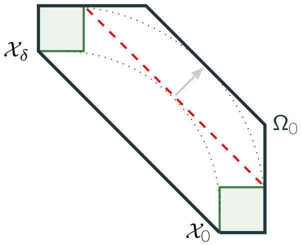
Given initial approximations Ω_0 and Ψ_δ such that

$$\bigcup_{0\leq t\leq\delta} \mathcal{X}_t \subseteq \Omega_0, \qquad \mathcal{Y}_\delta \subseteq \Psi_\delta,$$

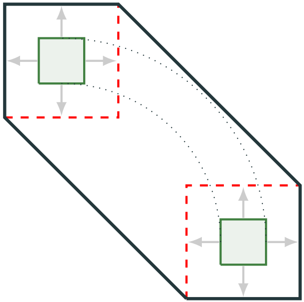
$\mathcal{X}_t$ is covered by the sequence

$$\Omega_{k+1} = e^{A\delta}\Omega_k \oplus \Psi_\delta.$$

Initial Approximations



(a) convex hull and pushing facets



(b) convex hull and bloating

Initial Approximations – Forward Bloating

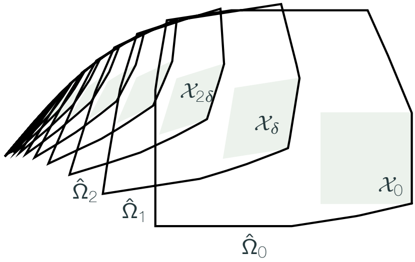
Bloating based on norms:[8]

$$\begin{aligned}
\Omega_0 &= \mathrm{chull}(\mathcal{X}_0 \cup e^{A\delta}\mathcal{X}_0) \oplus (\alpha_\delta + \beta_\delta)\mathcal{B}, \\
\Psi_\delta &= \beta_\delta \mathcal{B}, \\
\alpha_\delta &= \mu(\mathcal{X}_0) \cdot (e^{\|A\|\delta} - 1 - \|A\|\delta), \\
\beta_\delta &= \frac{1}{\|A\|}\mu(B\mathcal{U}) \cdot (e^{\|A\|\delta} - 1),
\end{aligned}$$

with radius $\mu(\mathcal{X}) = \max_{x \in \mathcal{X}} \|x\|$ and unit ball $\mathcal{B}$.

[8] A. Girard, "Reachability of uncertain linear systems using zonotopes," in *HSCC*, 2005, pp. 291–305.

Initial Approximations – Forward Bloating

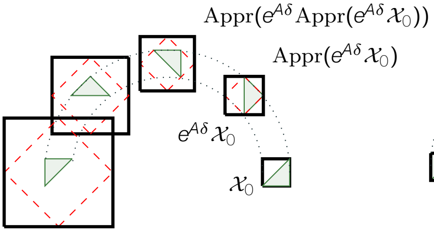


Forward bloating is tight on $\mathcal{X}_0$ and bloated on $\mathcal{X}_\delta$.

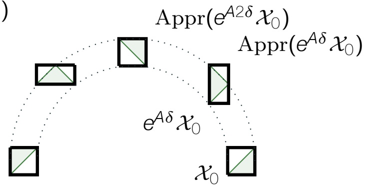
Improvements:

- intersect forward bloating with backward bloating
- bloat based on interpolation error (shown before)

Wrapping Effect



(a) with wrapping effect



(b) using a wrapping-free algorithm

avoid increasing complexity through approximation

$$\hat{\Omega}_{k+1} = \mathrm{Appr}(e^{A\delta}\hat{\Omega}_k \oplus \Psi_\delta).$$

wrapping effect: error accumulation

Wrapping Effect

Solution: Split sequence[9]

$$\begin{aligned}\hat{\Psi}_{k+1} &= \mathrm{Appr}(e^{Ak\delta}\Psi_\delta) \oplus \hat{\Psi}_k, \qquad \text{with } \hat{\Psi}_0 = \{0\},\\ \hat{\Omega}_k &= \mathrm{Appr}(e^{Ak\delta}\Omega_0) \oplus \hat{\Psi}_k.\end{aligned}$$

satisfies $\hat{\Omega}_k = \mathrm{Appr}(\Omega_k)$ (wrapping-free) if

$$\mathrm{Appr}(\mathcal{P} \oplus \mathcal{Q}) = \mathrm{Appr}(\mathcal{P}) \oplus \mathrm{Appr}(\mathcal{Q}),$$

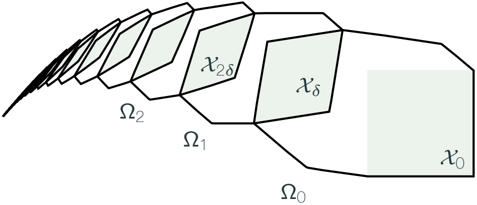
e.g., bounding box.

[9] A. Girard, C. L. Guernic, and O. Maler, "Efficient computation of reachable sets of linear time-invariant systems with inputs," in *HSCC*, 2006, pp. 257–271.

Overview

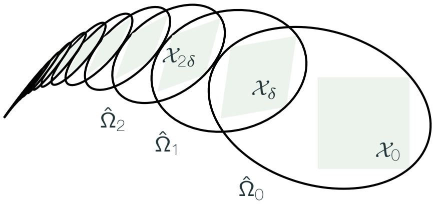
- Hybrid Automata
- Numerical Simulation
- Set-Based Reachability
  - Piecewise Constant Dynamics
  - Piecewise Affine Dynamics
  - Set Representations
  - SpaceEx (advertisement)
- Conclusions

Polyhedra



	polyhedra	
operation	m constr.	k gen.
convex hull	exp	$2k$
Minkowski sum	exp	k^2
linear map	m / exp	k
intersection	$2m$	exp

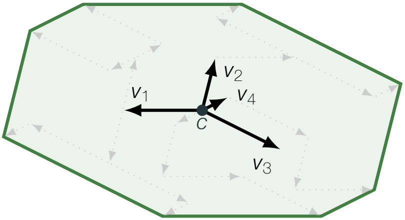
Ellipsoids[10]



	polyhedra		ellipsoids
operation	m constr.	k gen.	$n \times n$ matrix
convex hull	exp	$2k$	approx
Minkowski sum	exp	k^2	approx
linear map	m / exp	k	$\mathcal{O}(n^3)$
intersection	$2m$	exp	approx

[10] A. B. Kurzhanski and P. Varaiya, *Dynamics and Control of Trajectory Tubes*. Springer, 2014.

Zonotopes



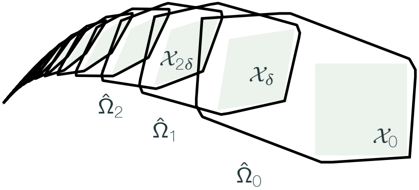
Zonotope with center $\mathbf{c} \in \mathbb{R}^n$ and generators $\mathbf{v}_1, \ldots, \mathbf{v}_k \in \mathbb{R}^n$

$$\mathcal{P} = \left\{ \mathbf{c} + \sum_{i=1}^{k} \alpha_i \mathbf{v}_i \;\middle|\; \alpha_i \in [-1, 1] \right\}.$$

linear map: map center and generators

Minkowski sum: add centers, take union of generators

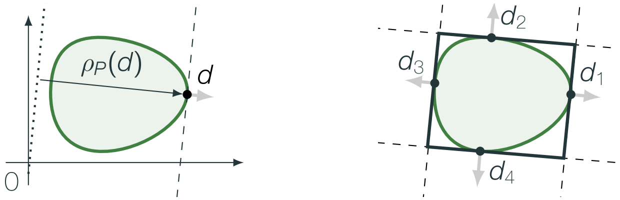
Zonotopes[11]



	polyhedra		ellipsoids	zonotopes
operation	m constr.	k gen.	$n \times n$ matrix	k generators
convex hull	exp	$2k$	approx	approx
Minkowski sum	exp	k^2	approx	$2k$
linear map	m / exp	k	$\mathcal{O}(n^3)$	k
intersection	$2m$	exp	approx	approx

[11] A. Girard, "Reachability of uncertain linear systems using zonotopes," in *HSCC*, 2005, pp. 291–305.

Support Functions



(a) support function in direction d (b) outer approximation

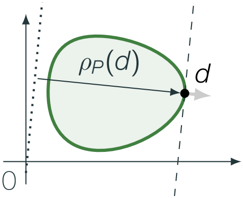
support function = linear optimization (efficient!)

$$\rho_{\mathcal{P}}(\mathbf{d}) = \max\{\mathbf{d}^{\mathsf{T}}\mathbf{x} \mid \mathbf{x} \in \mathcal{P}\}.$$

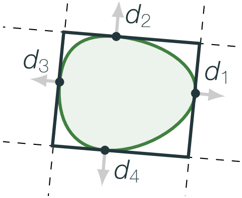
computed values define polyhedral **outer approximation**

$$\lceil \mathcal{P} \rceil_{\mathcal{D}} = \bigcap_{\mathbf{d} \in \mathcal{D}} \{\mathbf{d}^{\mathsf{T}} x \leq \rho_{\mathcal{P}}(\mathbf{d})\}.$$

Support Functions



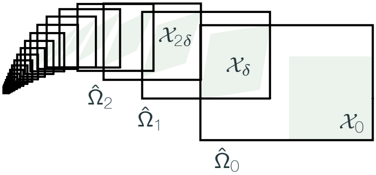
(a) support function in direction d

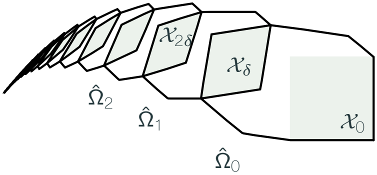


(b) outer approximation

- linear map: $\rho_{M\mathcal{X}}(\ell) = \rho_{\mathcal{X}}(M^\top \ell)$, $\mathcal{O}(mn)$,
- convex hull: $\rho_{\mathrm{chull}(\mathcal{P}\cup\mathcal{Q})}(\ell) = \max\{\rho_{\mathcal{P}}(\ell), \rho_{\mathcal{Q}}(\ell)\}$, $\mathcal{O}(1)$,
- Minkowski sum: $\rho_{\mathcal{X}\oplus\mathcal{Y}}(\ell) = \rho_{\mathcal{X}}(\ell) + \rho_{\mathcal{Y}}(\ell)$, $\mathcal{O}(1)$.

Support Functions (Le Guernic, Girard,'09)[13]



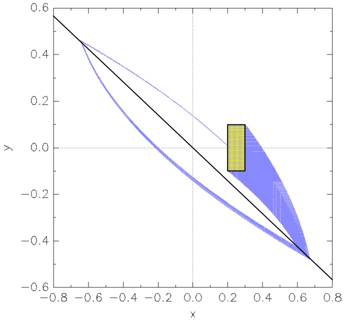


support functions: lazy approximation on demand

	polyhedra		ellipsoids	zonotopes	support f.
operation	m constr.	k gen.	$n \times n$ matrix	k generators	—
convex hull	exp	$2k$	approx	approx	$\mathcal{O}(1)$
Minkowski sum	exp	k^2	approx	$2k$	$\mathcal{O}(1)$
linear map	m / exp	k	$\mathcal{O}(n^3)$	k	$\mathcal{O}(n^2)$
intersection	$2m$	exp	approx	approx	opt. / approx

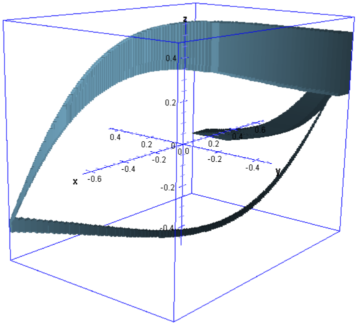
Example: Switched Oscillator

- **Switched oscillator**
  - 2 continuous variables
  - 4 discrete states
  - similar to many circuits (Buck converters,…)
- **plus linear filter**
  - m continuous variables
  - dampens output signal
- **affine dynamics**
  - total $2 + m$ continuous variables

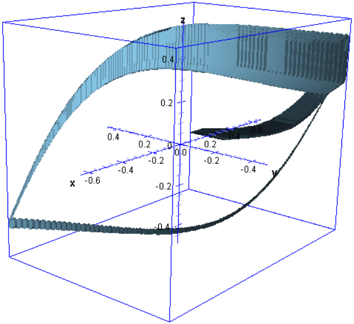


Example: Switched Oscillator

- **Low number of directions sufficient?**
  - here: 6 state variables



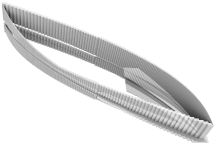
12 box constraints
(axis directions)

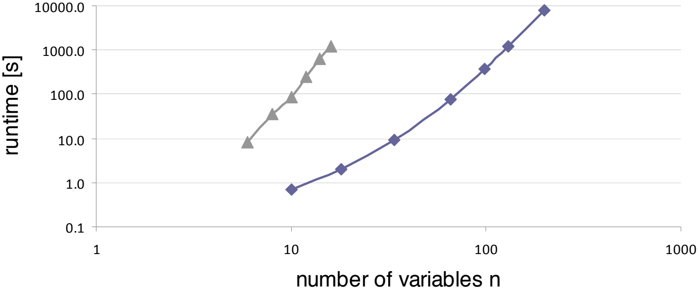


72 octagonal constraints
$(\pm x_i \pm x_j)$

Example: Switched Oscillator

- **Scalability Measurements:**
  - fixpoint reached in $\mathrm{O}(nm^2)$ time
  - box constraints: $\mathrm{O}(n^3)$
  - octagonal constraints: $\mathrm{O}(n^5)$





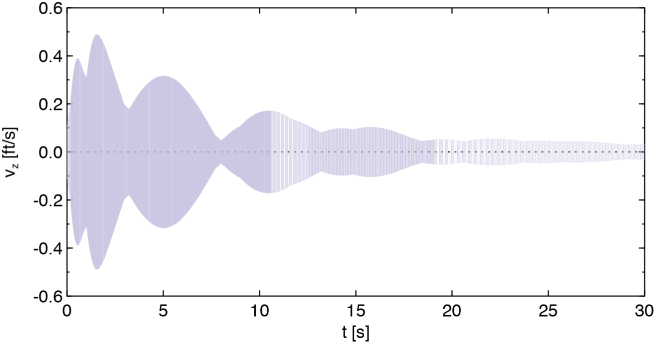
Example: Controlled Helicopter

Photo by Andrew P Clarke

- **28-dim model of a Westland Lynx helicopter**
  - 8-dim model of flight dynamics
  - 20-dim continuous H∞ controller for disturbance rejection
  - stiff, highly coupled dynamics

Example: Helicopter

- **28 state variables + clock**

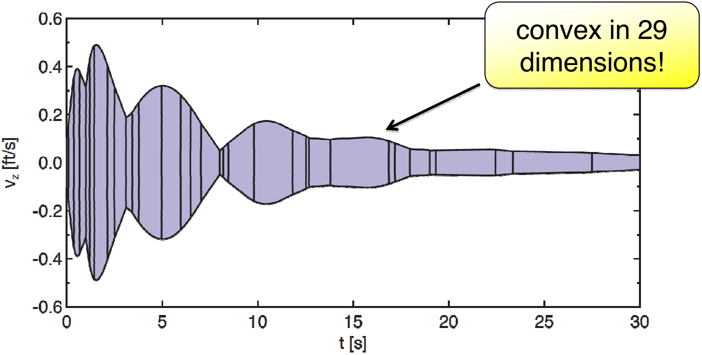


CAV’11: 1440 sets in 5.9s

1440 time steps

Example: Helicopter

- **28 state variables + clock**

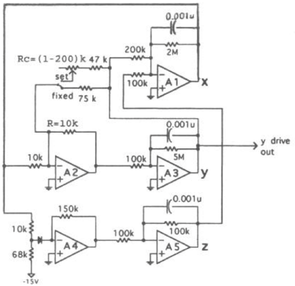


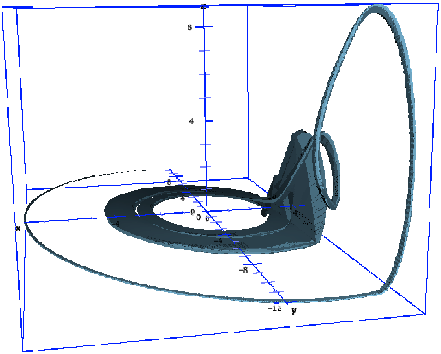
HSCC’13: **32** sets in 15.2s (4.8s clustering)

2 -- 3300 time steps, median 360

Example: Chaotic Circuit

- **piecewise linear Rössler-like circuit**
  Pisarchik, Jaimes-Reátegui. ICCSDS'05
- **added nondet. disturbances**
- **3 variables, hard!**





Nonlinear Dynamics – Linearization

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

with f globally Lipschitz continuous.

Linearization: choose domain $\mathcal{S}$ (partition, sliding window)

overapproximate in $\mathcal{S}$ with $\dot{\mathbf{x}} = A\mathbf{x} + \mathbf{u}, \mathbf{u} \in \mathcal{U}$

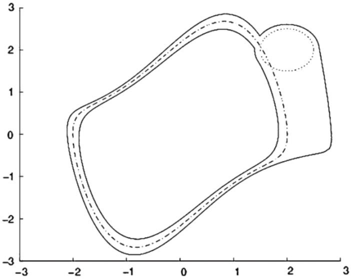
linearizing $f(x)$ around $\mathbf{x}_0 \in \mathcal{S}$ gives

$$a_{ij} = \left.\frac{\partial f_i}{\partial x_j}\right|_{\mathbf{x}=\mathbf{x}_0} \text{ and } \mathbf{b} = f(\mathbf{x}_0) - A\mathbf{x}_0.$$

$$\mathcal{U} = \mathrm{Appr}\left\{f(\mathbf{x}) - (A\mathbf{x} + \mathbf{b}), \mathbf{x} \in \mathcal{S}\right\} \oplus \mathbf{b}.$$

Example: Van der Pol Oscillator[12]

$$
\begin{aligned}
\dot{x} &= y \\
\dot{y} &= y(1 - x^2) - x
\end{aligned}
$$



hybridization: here triangular partition of size 0.05

partitioning generally doesn't scale well

[12] E. Asarin, T. Dang, and A. Girard, "Hybridization methods for the analysis of nonlinear systems," *Acta Inf.*, vol. 43, no. 7, pp. 451–476, 2007.

Nonlinear Dynamics – Polynomial Approximations

Bernstein polynomials for polynomial $f(\mathbf{x})$

- polyhedral approximation of successors[13]

Taylor models

- polynomial approximations of Taylor expansion
- represent sets with polynomials
- **Flow*** verification tool[16]

[13] T. Dang and R. Testylier, "Reachability analysis for polynomial dynamical systems using the bernstein expansion," *Reliable Computing*, vol. 17, no. 2, pp. 128–152, 2012.

Overview

Hybrid Automata

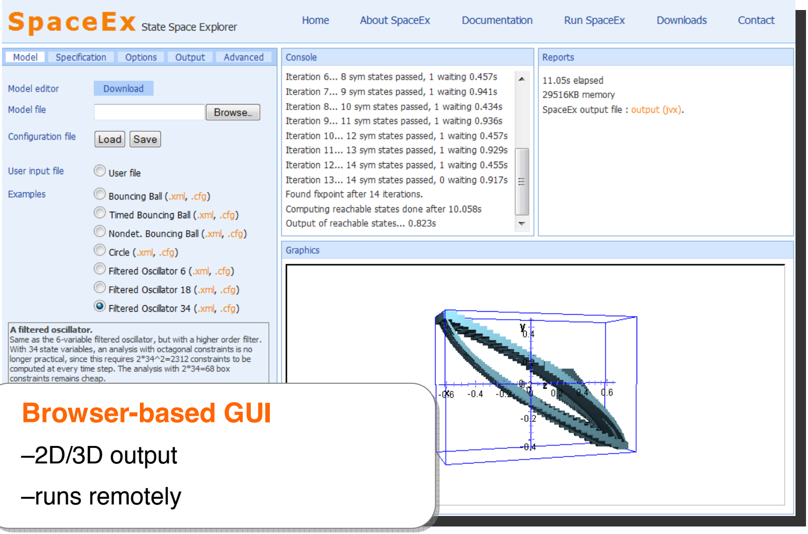
Numerical Simulation

Set-Based Reachability

- Piecewise Constant Dynamics
- Piecewise Affine Dynamics
- Set Representations
- SpaceEx (advertisement)

Conclusions

SpaceEx Verification Platform

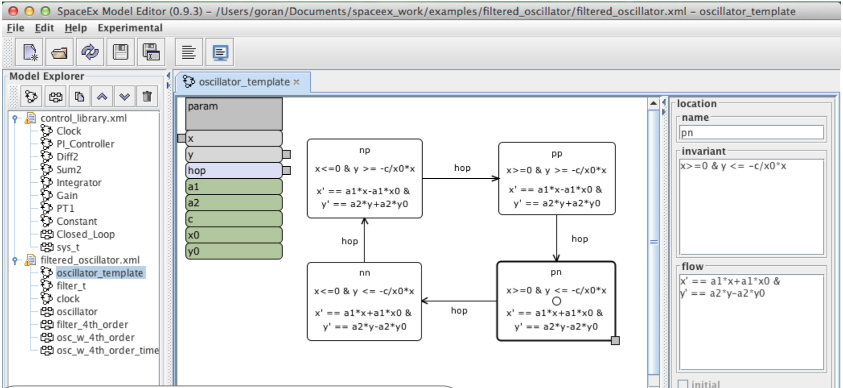


Browser-based GUI

–2D/3D output

–runs remotely

SpaceEx Model Editor

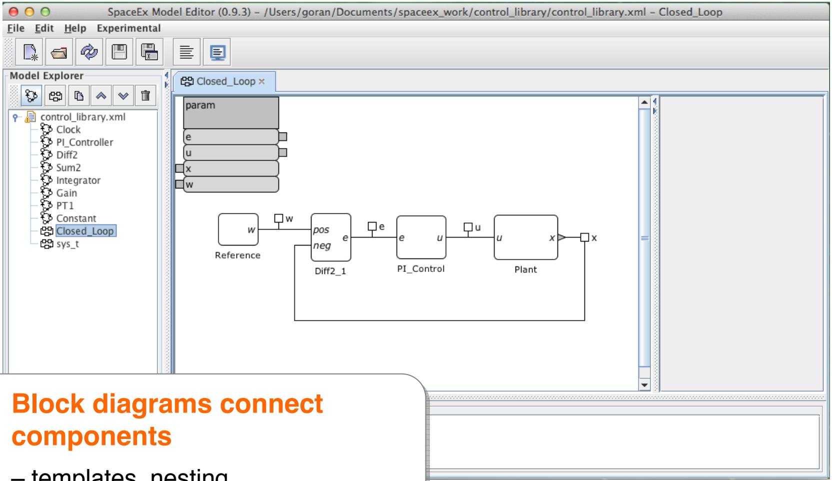


Components = Hybrid Automata

– real-values variables

– ODE, linear DAE

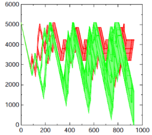
SpaceEx Model Editor



Block diagrams connect components

– templates, nesting

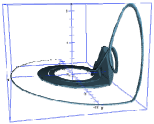
SpaceEx Reachability Algorithms



PHAVer

–constant dynamics (LHA)

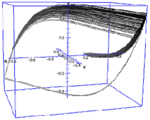
–formally sound and exact



Support Function Algo

–many continuous variables

–low discrete complexity



Simulation

–nonlinear dynamics

–based on CVODE

spaceex.imag.fr

Overview

Hybrid Automata

Numerical Simulation

Set-Based Reachability

Conclusions

Conclusions

- Hybrid systems are easy to model with **hybrid automata** but difficult to analyze.
- **Numerical simulation** scales, but is not exhaustive and critical behavior may be missed.
- **Set-based reachability** covers all runs, sufficient for safety and bounded liveness.
  - computational cost,
  - scalable for piecewise affine dynamics
- Remaining challenges: trade-off between approximation accuracy and computational cost, scalable extension to nonlinear dynamics

References I

[2] R. Alur, C. Courcoubetis, N. Halbwachs, T. Henzinger, P.-H. Ho, X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine, “The algorithmic analysis of hybrid systems,” *Theoretical Computer Science*, vol. 138, pp. 3–34, 1995.

[3] T. A. Henzinger, “The theory of hybrid automata.,” in *LICS*, Los Alamitos: IEEE Computer Society, 1996, pp. 278–292.

[13] C. Le Guernic and A. Girard, “Reachability analysis of linear systems using support functions,” *Nonlinear Analysis: Hybrid Systems*, vol. 4, no. 2, pp. 250–262, 2010.

[16] X. Chen, E. Ábrahám, and S. Sankaranarayanan, “Taylor model flowpipe construction for non-linear hybrid systems,” in *RTSS*, IEEE Computer Society, 2012, pp. 183–192, ISBN: 978-1-4673-3098-5.