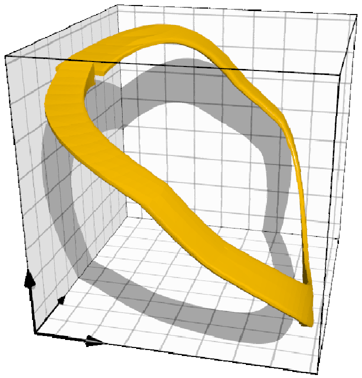
Tools for Hybrid Systems Reachability



Goran Frehse
Universite Grenoble 1, Verimag

- with work from Thao Dang, Antoine Girard and Colas Le Guernic -

QMC'10, Copenhague, March 5, 2010

Outline

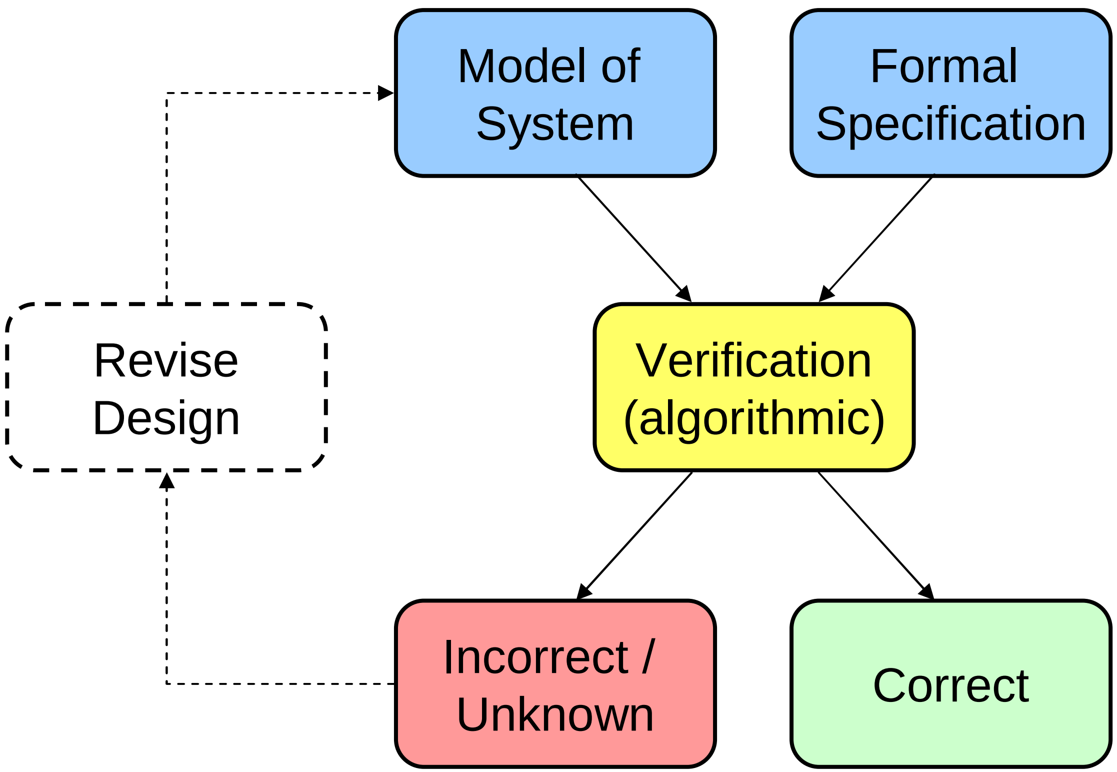
I. Hybrid Automata and Reachability

II. Linear Hybrid Automata

III. Piecewise Affine Hybrid Systems

IV. Support Functions

Formal Verification



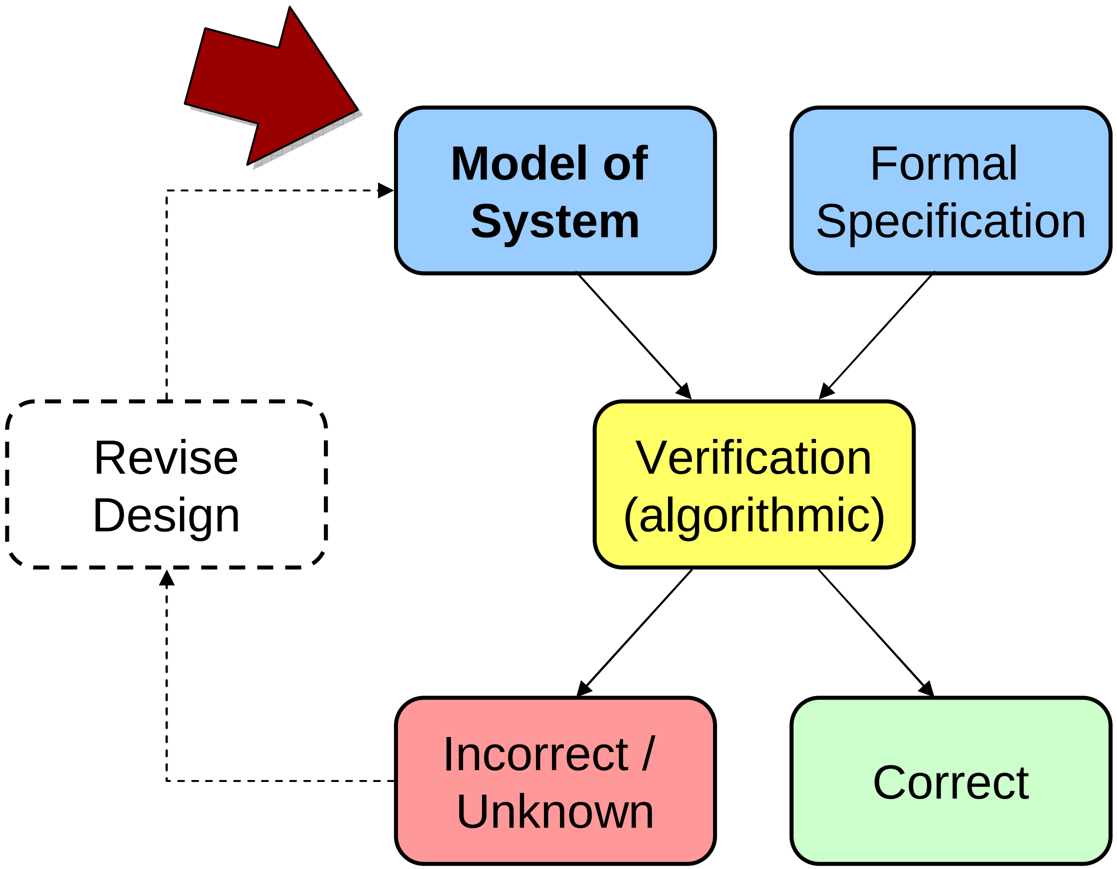
Formal Verification

- **Key Problems**
  - computable (decidable) only for simple dynamics
  - computationally expensive
  - representation of / computation with continuous sets

Formal Verification

- **Fighting complexity with overapproximations**
  - simplify dynamics
  - set representations
  - set computations
- **Overapproximations should be**
  - conservative
  - easy to derive and compute with
  - accurate (not too many false positives)

Formal Verification



Formal Verification

Model of System

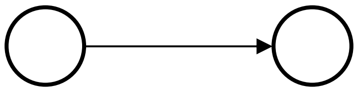
Model of Physics

Model of Software

continuous dynamics

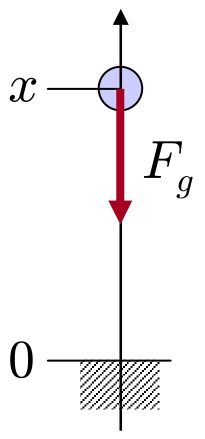
$$\dot{x} = f(x)$$

discrete dynamics



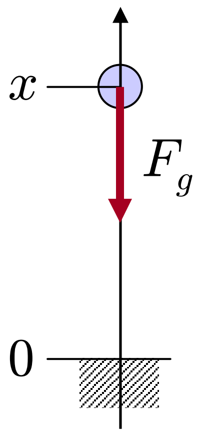
Modeling Hybrid Systems

- **Example: Bouncing Ball**
  - ball with mass m and position x in free fall
  - bounces when it hits the ground at $x = 0$
  - initially at position x_0 and at rest



Part I – Free Fall

- **Condition for Free Fall**
  - ball above ground: $x \geq 0$



- **First Principles (physical laws)**
  - gravitational force :

$$F_g = -mg$$

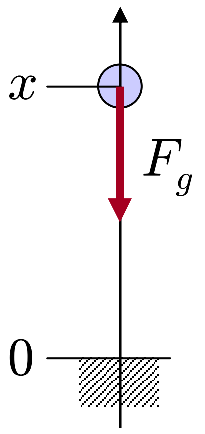
$$g = 9.81\mathrm{m/s}^2$$

  - Newton's law of motion :

$$m\ddot{x} = F_g$$

Part I – Free Fall

$$\begin{aligned} F_g &= -mg \\ m\ddot{x} &= F_g \end{aligned}$$



- **Obtaining 1st Order ODE System**
  - ordinary differential equation $\dot{x} = f(x)$
  - transform to 1st order by introducing variables for higher derivatives
  - here: $v = \dot{x}$:

$$\begin{aligned} \dot{x} &= v \\ \dot{v} &= -g \end{aligned}$$

Part II – Bouncing

- **Conditions for "Bouncing"**
  - ball at ground position: $x = 0$
  - downward motion: $v < 0$
- **Action for "Bouncing"**
  - velocity changes direction
  - loss of velocity (deformation, friction)
  - $v := -cv$, $0 \leq c \leq 1$

Combining Part I and II

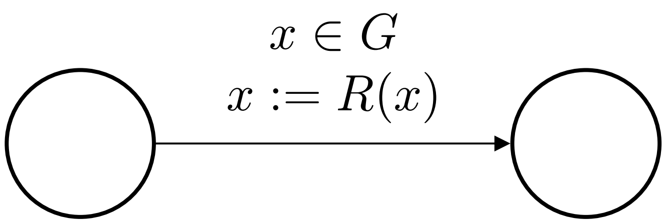
- **Free Fall**
  - while $x \geq 0$,
    $$\begin{aligned} \dot{x} &= v \\ \dot{v} &= -g \end{aligned}$$

continuous dynamics

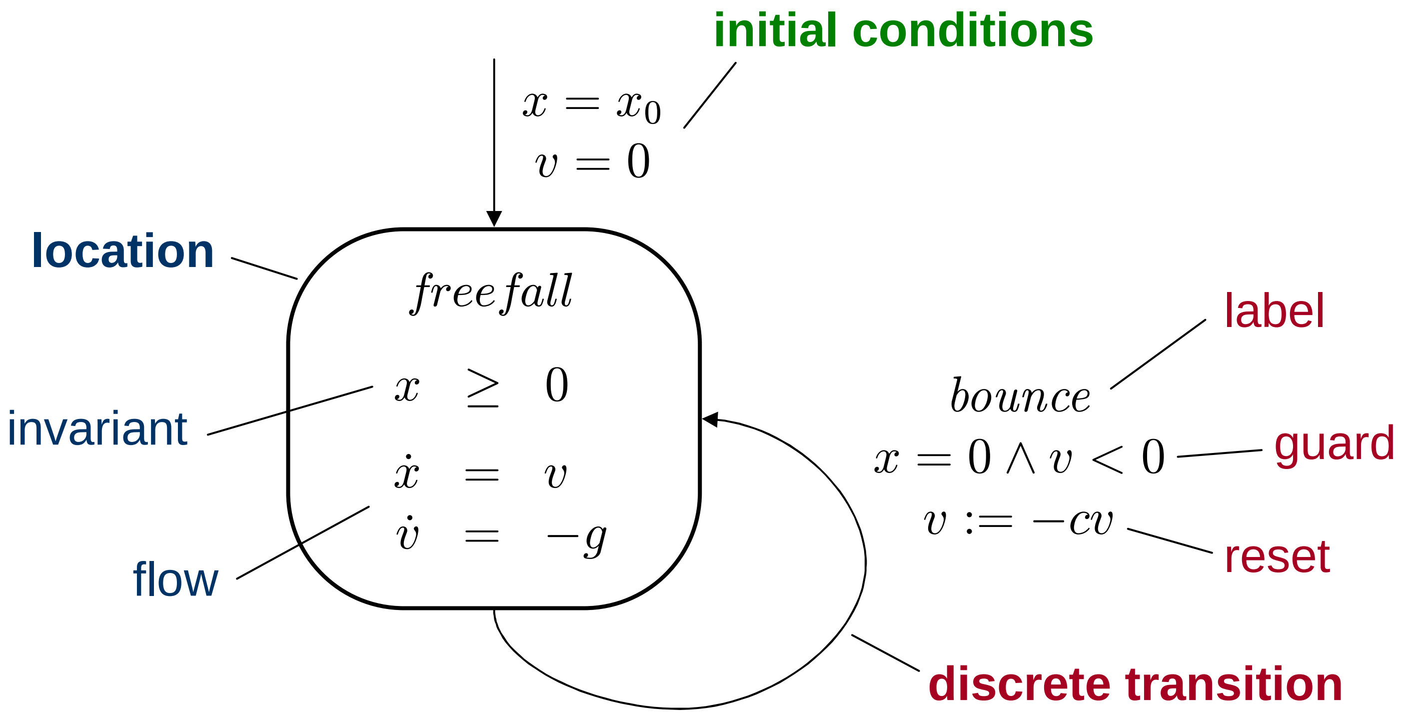
$$\dot{x} = f(x)$$

- **Bouncing**
  - if $x = 0$ and $v < 0$
    $$v := -cv$$

discrete dynamics

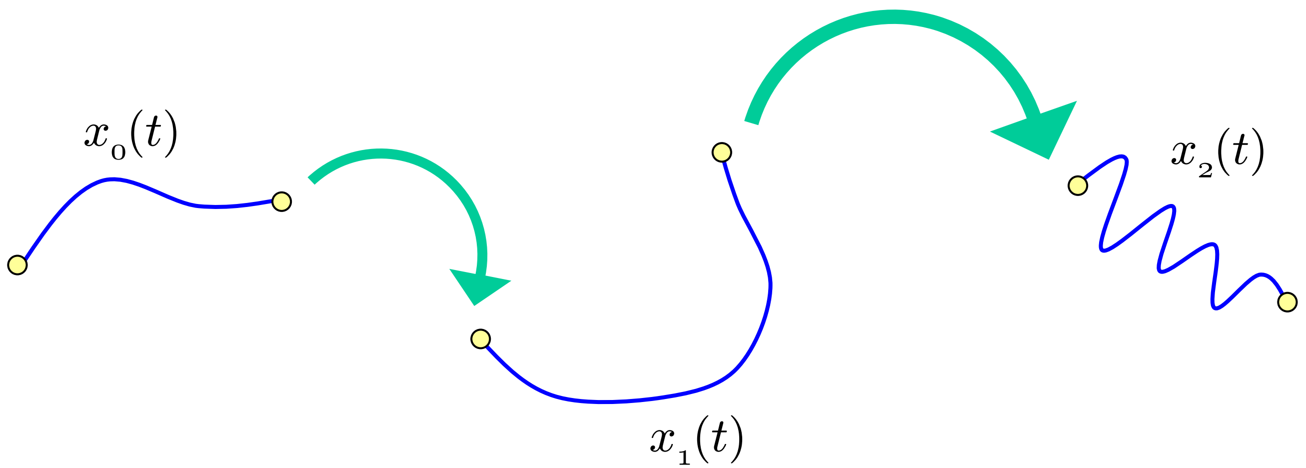


Hybrid Automaton Model

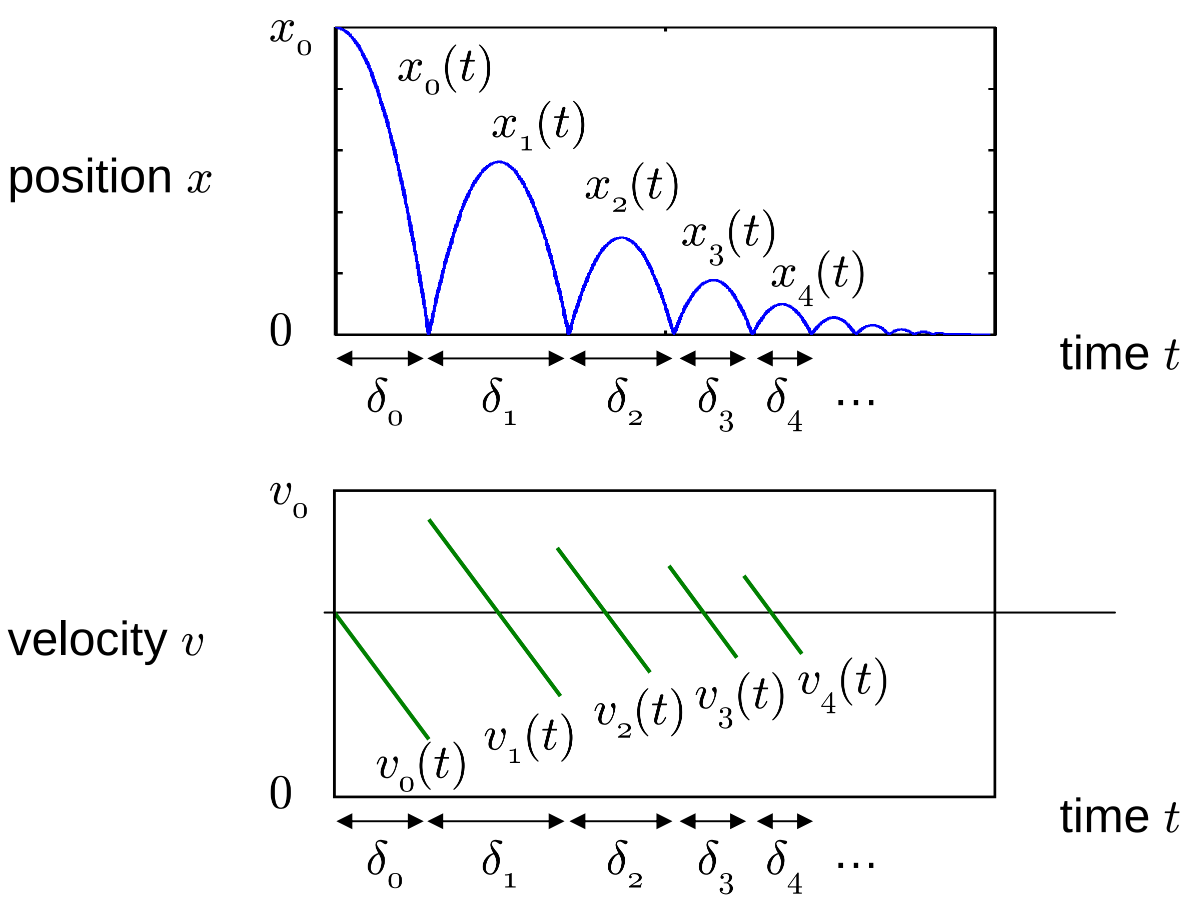


Hybrid Automata - Semantics

- **Run**
  - sequence of discrete transitions and time elapse
- **Execution**
  - run that starts in the initial states

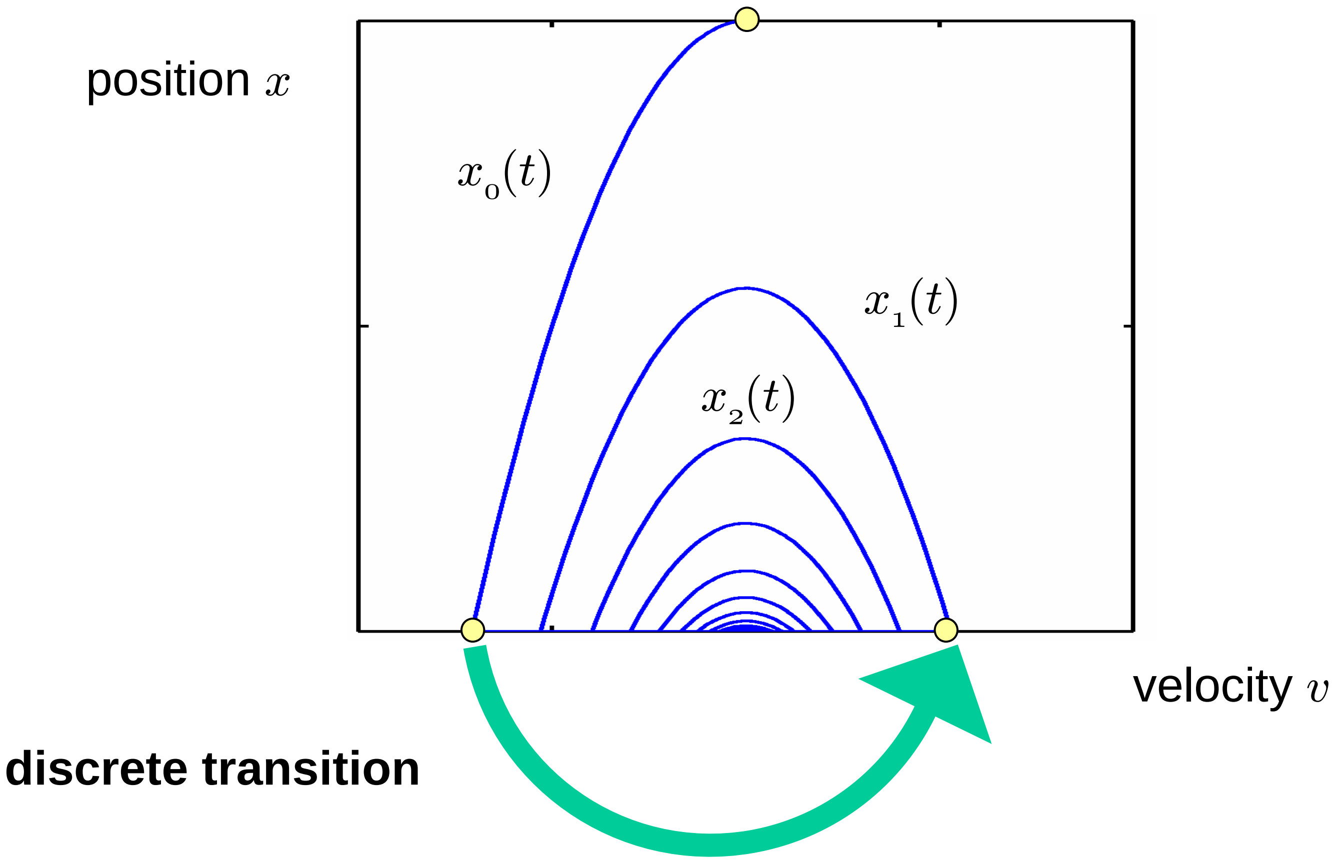


Execution of Bouncing Ball

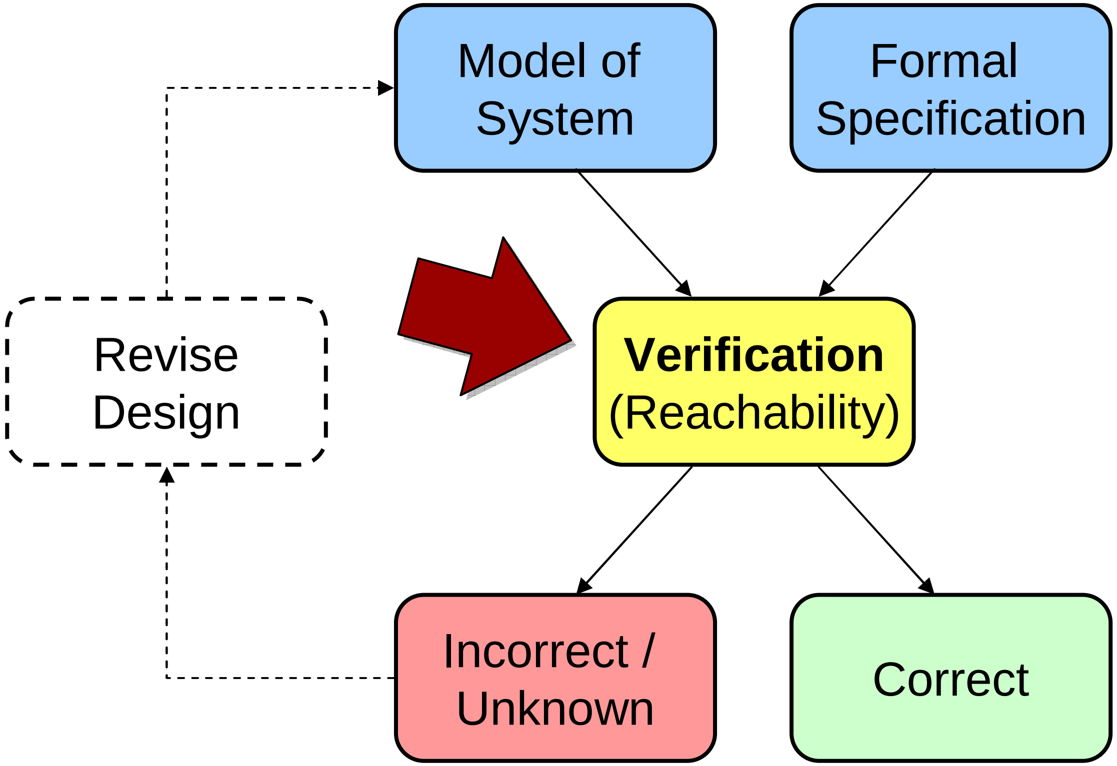


Execution of Bouncing Ball

- **State-Space View (infinite time range)**

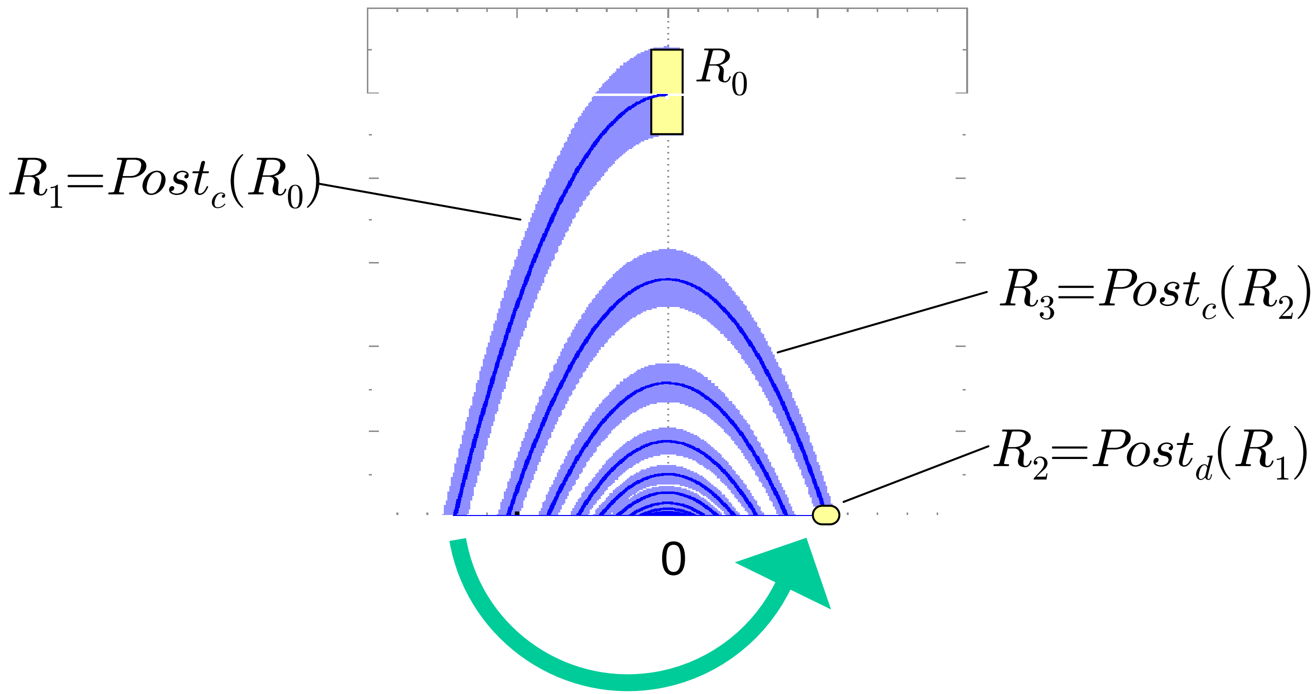


Formal Verification



Computing Reachable States

- **Compute successor states**
  - discrete transitions : $Post_d(R)$
  - time elapse : $Post_c(R)$



Computing Reachable States

- **Fixpoint computation**
  - Initialization: $R_0 = Ini$
  - Recurrence: $R_{k+1} = R_k \cup Post_d(R_k) \cup Post_c(R_k)$
  - Termination: $R_{k+1} = R_k \Rightarrow Reach = R_k$.
- **Problems**
  - in general termination not guaranteed
  - time-elapse very hard to compute with sets

Chapter Summary

- **Why should we care?**
  - Reachability Analysis is a set-based computation that can answer many interesting questions about a system (safety, bounded liveness,…)
- **What's the problem?**
  - The hardest part is computing time elapse.
  - Explicit solutions only for very simple dynamics.
- **What's the solution?**
  - First study simple dynamics.
  - Then apply these techniques to complex dynamics.

Outline

I. Hybrid Automata and Reachability

II. Linear Hybrid Automata

III. Piecewise Affine Hybrid Systems

IV. Support Functions

In this Chapter…

- **A very simple class of hybrid systems**
- **Exact computation of discrete transitions and time elapse**
  - Note: Reachability (and pretty much everything else) is nonetheless **undecidable**.
- **A case study**

Linear Hybrid Automata

- **Continuous Dynamics**
  - piecewise constant: $\dot{x} = 1$
  - intervals: $\dot{x} \in [1, 2]$
  - conservation laws: $\dot{x}_1 + \dot{x}_2 = 0$
  - general form: conjunctions of linear constraints

$$a \cdot \dot{x} \bowtie b, \qquad a \in \mathbb{Z}^n, b \in \mathbb{Z}, \bowtie \in \{<, \leq\}.$$

= convex polyhedron over derivatives

Linear Hybrid Automata

- **Discrete Dynamics**
  - affine transform: $x := ax + b$
  - with intervals: $x_2 := x_1 \pm 0.5$
  - general form: conjunctions of linear constraints (new value x')

$$a \cdot x + a' \cdot x' \bowtie b, \qquad a, a' \in \mathbb{Z}^n, b \in \mathbb{Z}, \bowtie \in \{<, \leq\}$$

= convex polyhedron over x and x'

Linear Hybrid Automata

- **Invariants, Initial States**
  - general form: conjunctions of linear constraints

$$a \cdot x \bowtie b, \qquad a \in \mathbb{Z}^n, b \in \mathbb{Z}, \bowtie \in \{<, \leq\},$$

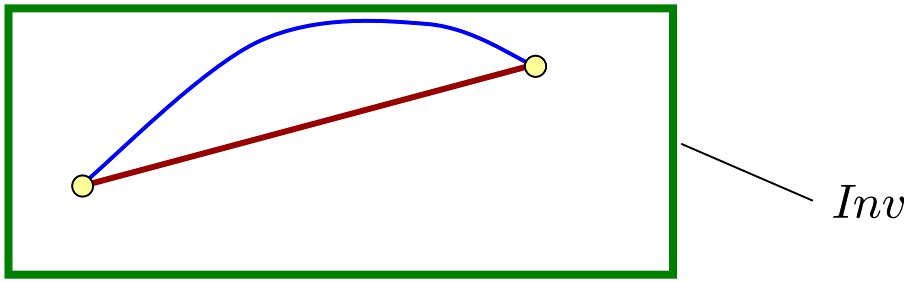
= convex polyhedron over x

Reachability with LHA

- **Compute discrete successor states** $Post_d(S)$
  - all x' for which exists $x \in S$ s.t.
    - $x \in G$
    - x' $\in R(x) \cap Inv$
- **Operations:**
  - existential quantification
  - intersection
  - standard operations on convex polyhedra, but of exponential complexity

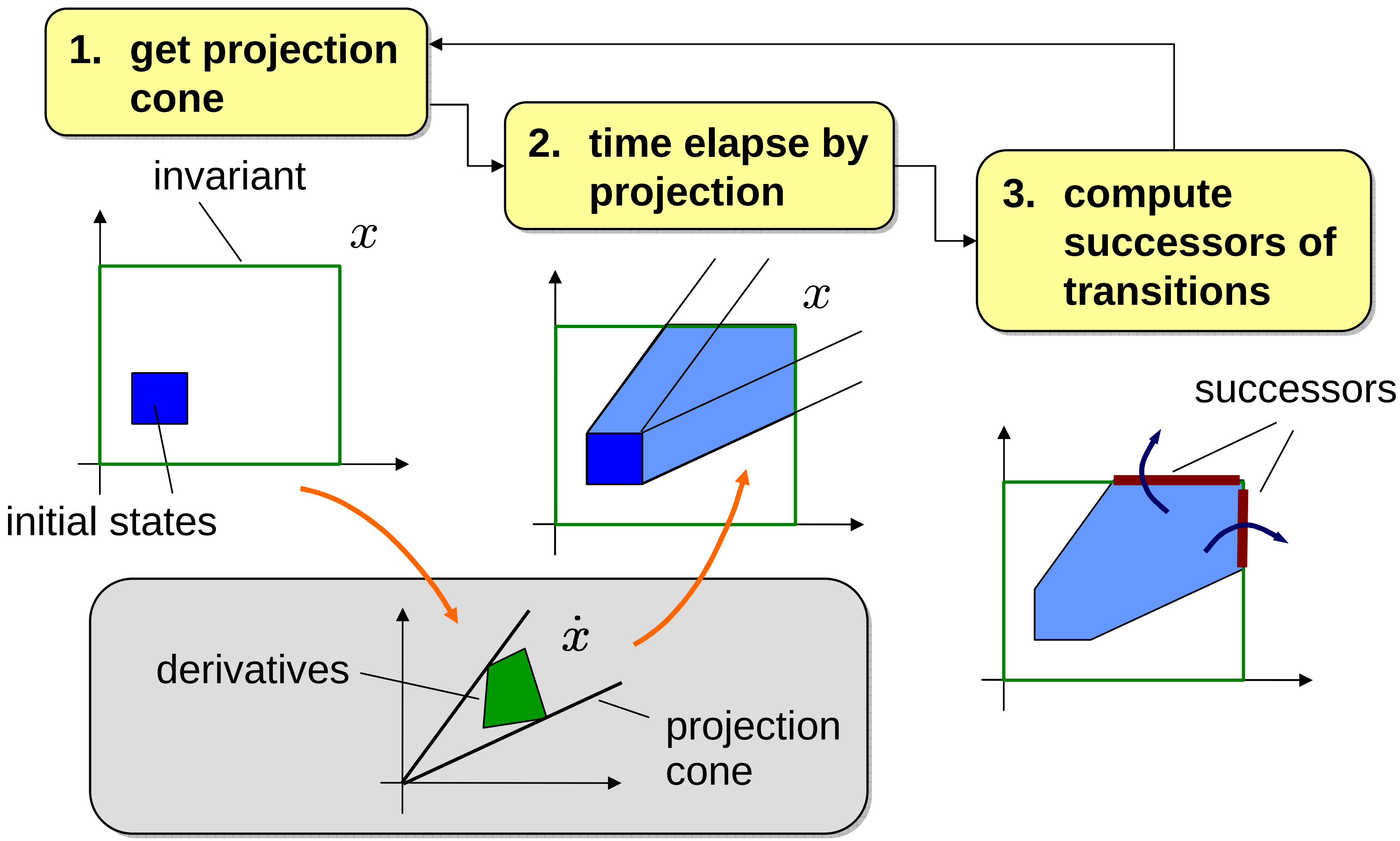
Reachability with LHA

- **Compute time elapse states** $Post_c(S)$
- **Theorem** [Alur et al.]
  - Time elapse along arbitrary trajectory iff time elapse along straight line (convex invariant).



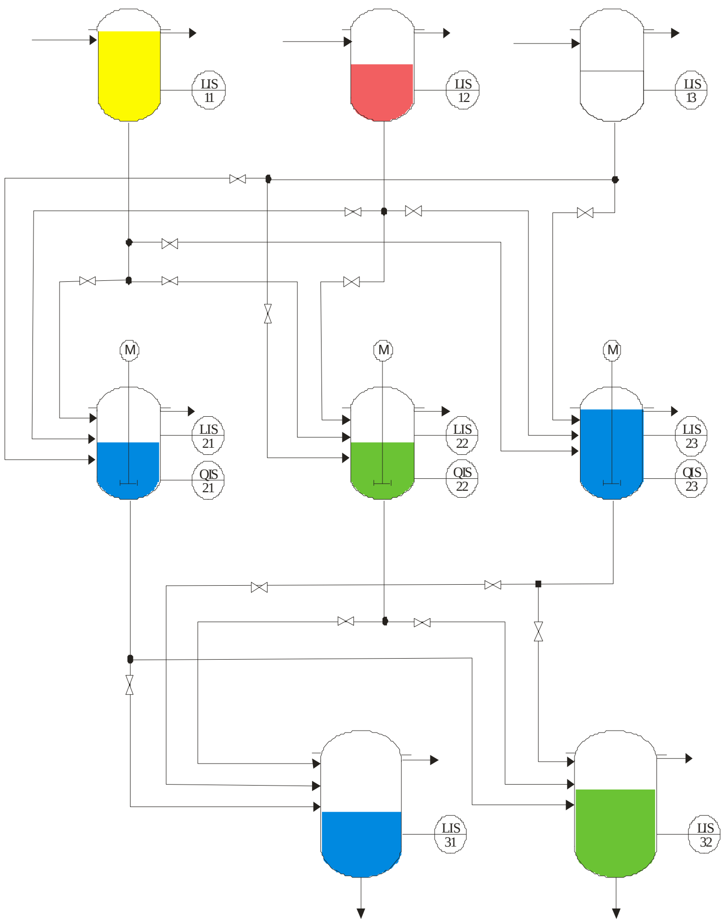
  - time elapse along straight line can be computed as projection along cone [Halbwachs et al.]

Reachability with LHA [Halbwachs, Henzinger, 93-97]



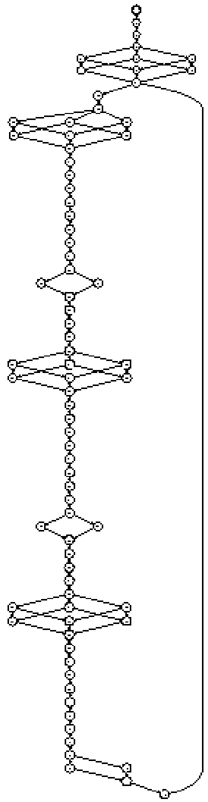
Multi-Product Batch Plant

Multi-Product Batch Plant

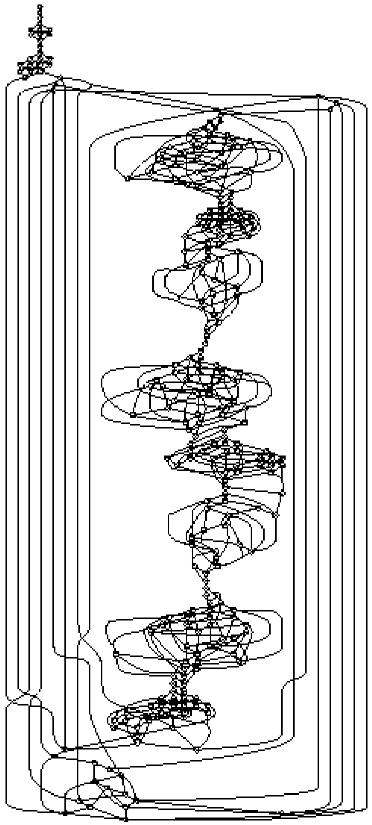


- **Cascade mixing process**
  - 3 educts via 3 reactors
    $\Rightarrow$ 2 products
- **Verification Goals**
  - Invariants
    - overflow
    - product tanks never empty
  - Filling sequence
- **Design of verified controller**

Verification with PHAVer



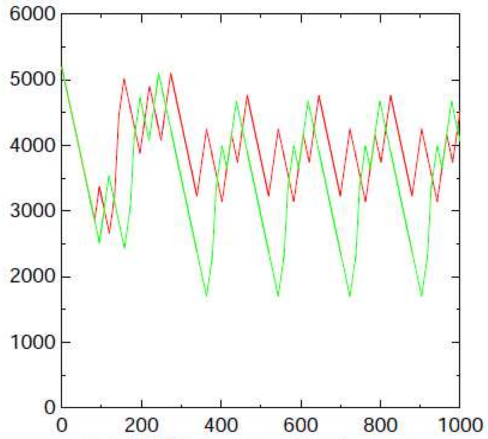
Controller



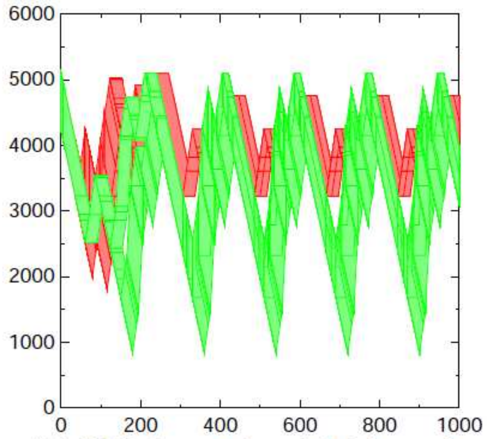
Controlled Plant

- **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)

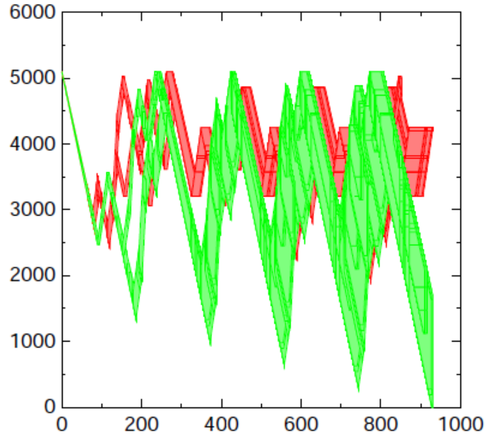
Verification with PHAVer



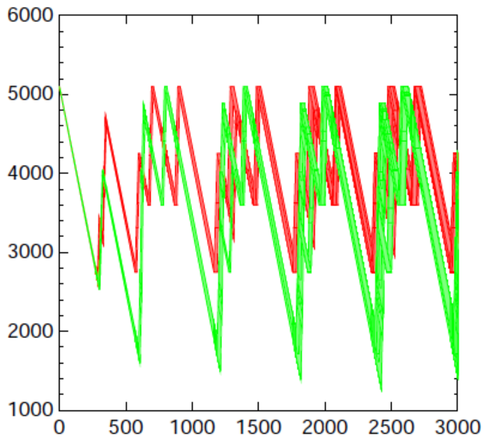
(a) BP8.1: nominal case



(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

Outline

I. Hybrid Automata and Reachability

II. Linear Hybrid Automata

III. Piecewise Affine Hybrid Systems

IV. Support Functions

In this Chapter…

- **Another class of (not quite so) simple dynamics**
  – but things are getting serious (no explicit solution for sets)
- **Exact computation of time elapse only at discrete points in time**
  – used to overapproximate continuous time
- **Efficient data structures**

Piecewise Affine Hybrid Systems

- **Affine dynamics**
  - Flow:

    $\dot{x} = Ax + b$ (deterministic)

    $\dot{x} \in Ax + B$, with B a set (nondeterministic)

  - For time elapse it's enough to look at a single location.

Linear Dynamics

- **Let's begin with "autonomous" part of the dynamics:**

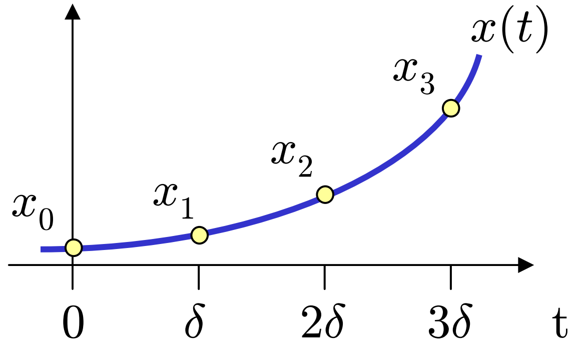
$$\dot{x} = Ax, \quad x \in \mathbb{R}^n$$

- **Known solutions:**
  - analytic solution in continuous time
  - explicit solution at discrete points in time (up to arbitrary accuracy)
- **Approach for Reachability:**
  - Compute reachable states over finite time: $Reach_{[0,\mathrm{T}]}(X_{Ini})$
  - Use time-discretization, but with care!

Time-Discretization for an Initial Point

- **Analytic solution:** $x(t) = e^{At} x_{Ini}$
  - with $t = \delta k$:

$$x(\delta(k+1)) = e^{A\delta} x(\delta k)$$



- **Explicit solution in discretized time (recursive):**

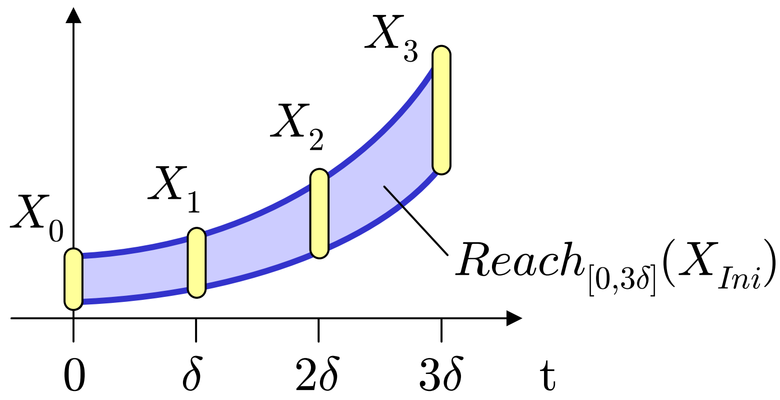
$$\begin{aligned} x_0 &= x_{Ini} \\ x_{k+1} &= e^{A\delta} x_k \end{aligned}$$

multiplication with const. matrix $e^{A\delta}$
= linear transform

Time-Discretization for an Initial Set

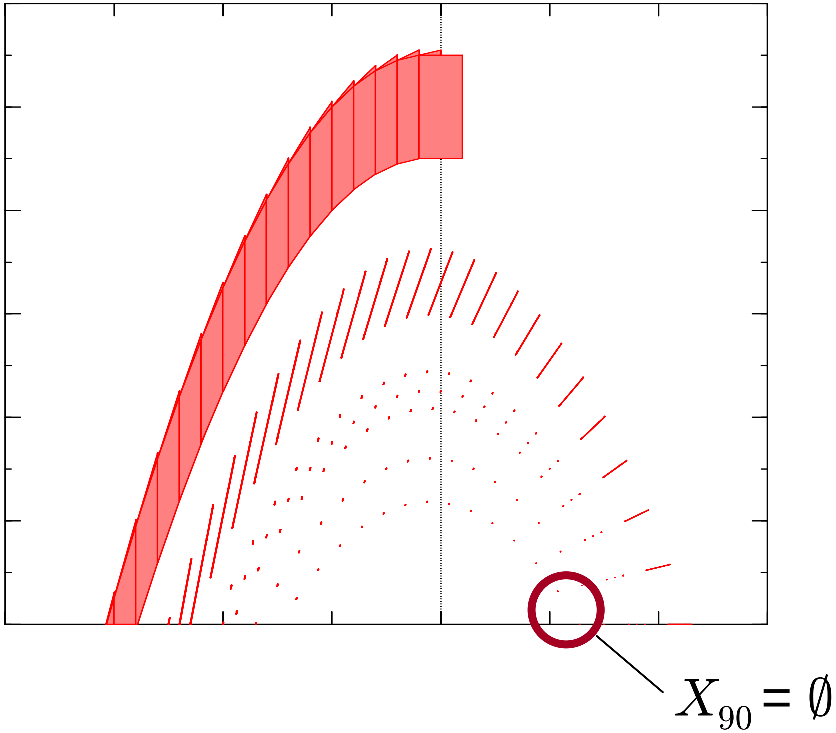
- **Explicit solution in discretized time**

$$\begin{aligned} X_0 &= X_{Ini} \\ X_{k+1} &= e^{A\delta} X_k \end{aligned}$$



- **Acceptable solution for purely continuous systems**
  - $x(t)$ is in $\epsilon(\delta)$-neighborhood of some X_k
- **Unacceptable for hybrid systems**
  - discrete transitions might “fire” between sampling times
  - if transitions are “missed,” $x(t)$ not in $\epsilon(\delta)$-neighborhood

Bouncing Ball



– In other examples this error might not be as obvious…

Reachability by Time-Discretization

- **Goal:**
  - Compute sequence Ω_k over bounded time $[0,N\delta]$ such that:

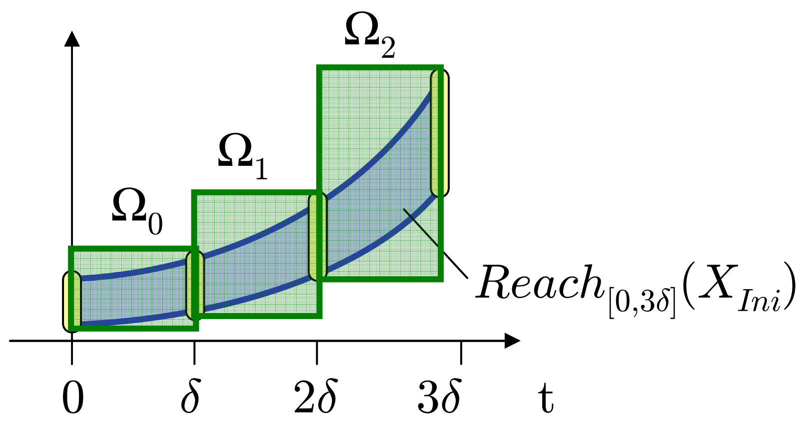
$$\mathrm{Reach}_{[0,N\delta]}(X_{Ini}) \subseteq \Omega_0 \cup \Omega_1 \cup \ldots \cup \Omega_N$$

- **Approach:**
  - Refine Ω_k by recurrence:

$$\Omega_{k+1} \quad = \quad e^{A\delta}\Omega_k$$

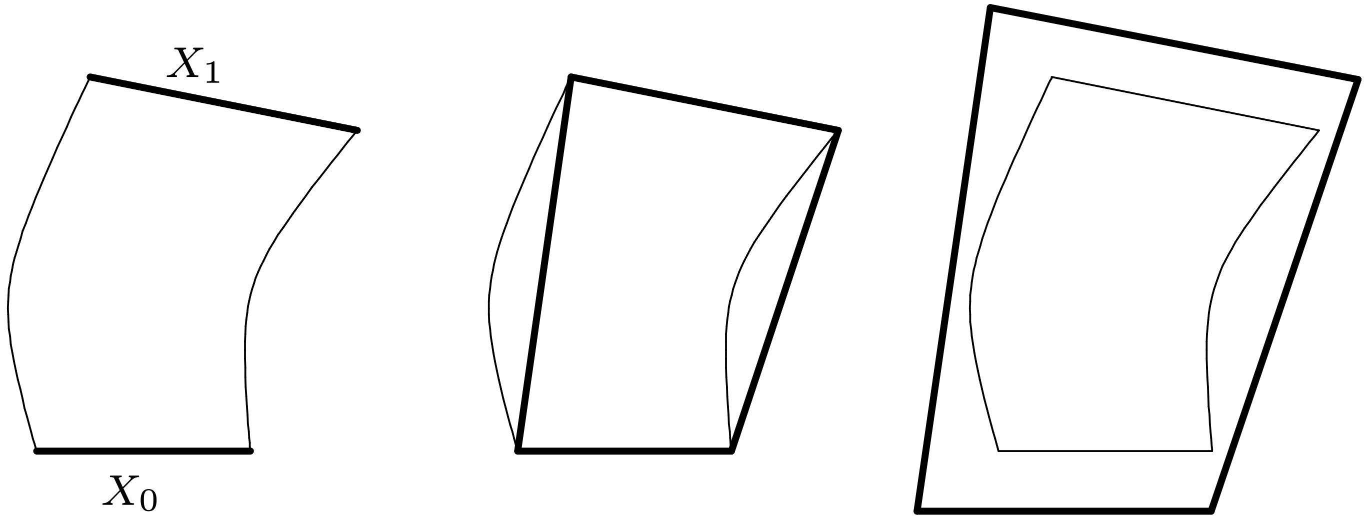
  - Condition for Ω_0:

$$\mathrm{Reach}_{[0,\delta]}(X_{Ini}) \subseteq \Omega_0$$



Time-Discretization with Convex Hull

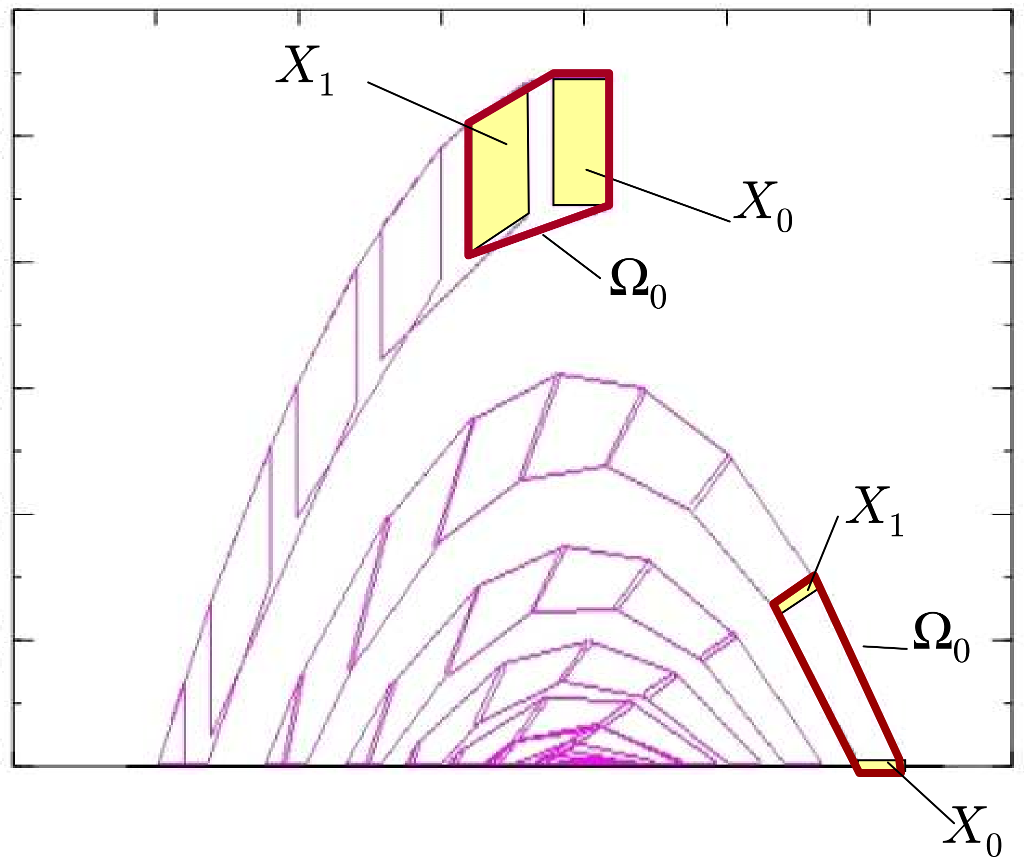
- **Overapproximating** $Reach_{[0,\delta]}$**:**



$\mathrm{Reach}_{[0,\delta]}(X_{Ini})$ $Conv(X_0, X_1)$ $Bloat(Conv(X_0, X_1))$

Time-Discretization with Convex Hull

- **Bouncing Ball:**



Nondeterministic Affine Dynamics

- **Let's include the effect of inputs:**

$$\dot{x} = Ax + Bu, \quad x \in \mathbb{R}^n, u \in U \subseteq \mathbb{R}^p$$

  – variables $x_1, \ldots, x_n$, inputs $u_1, \ldots, u_p$

- **Input u models nondeterminism**

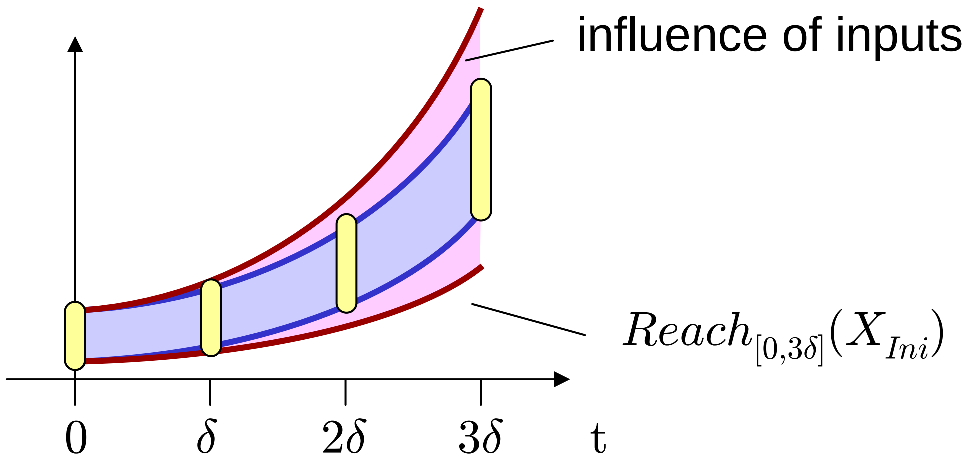
$$\dot{x} \in Ax + BU$$

  – used later for overapproximating nonlinear dynamics

Nondeterministic Affine Dynamics

- **Analytic Solution**

$$\underbrace{x(t) = e^{A\delta}x(0)}_{\text{autonomous dynamics}} + \underbrace{\int_0^{\tau} e^{A(\delta-\tau)}Bu(\tau)d\tau}_{\text{influence of inputs}}$$

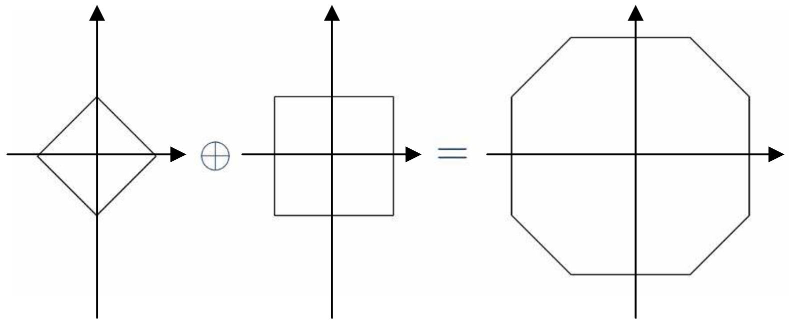


Nondeterministic Affine Dynamics

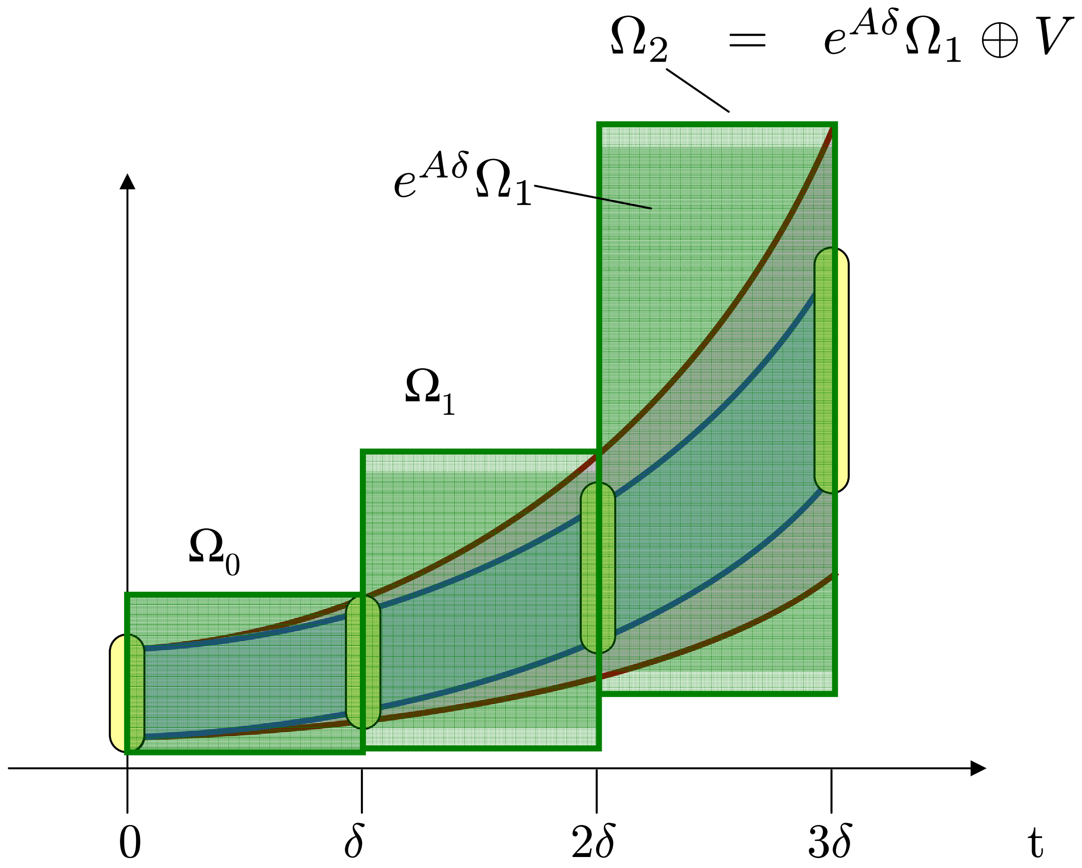
- **How far can the input "push" the system in δ time?**
  - $V = \text{box with radius } \frac{e^{||A||\delta}-1}{||A||} \sup_{u \in U} ||Bu||$

$$
\begin{aligned}
\Omega_0 &= Bloat(Conv(X_{Ini}, e^{A\delta} X_{Ini})) \oplus V \\
\Omega_{k+1} &= e^{A\delta}\Omega_k \oplus V
\end{aligned}
$$

- **Minkowski Sum:** $A \oplus B = \{a + b \mid a \in A,\ b \in B\}$



Nondeterministic Affine Dynamics



Wrapping Effect

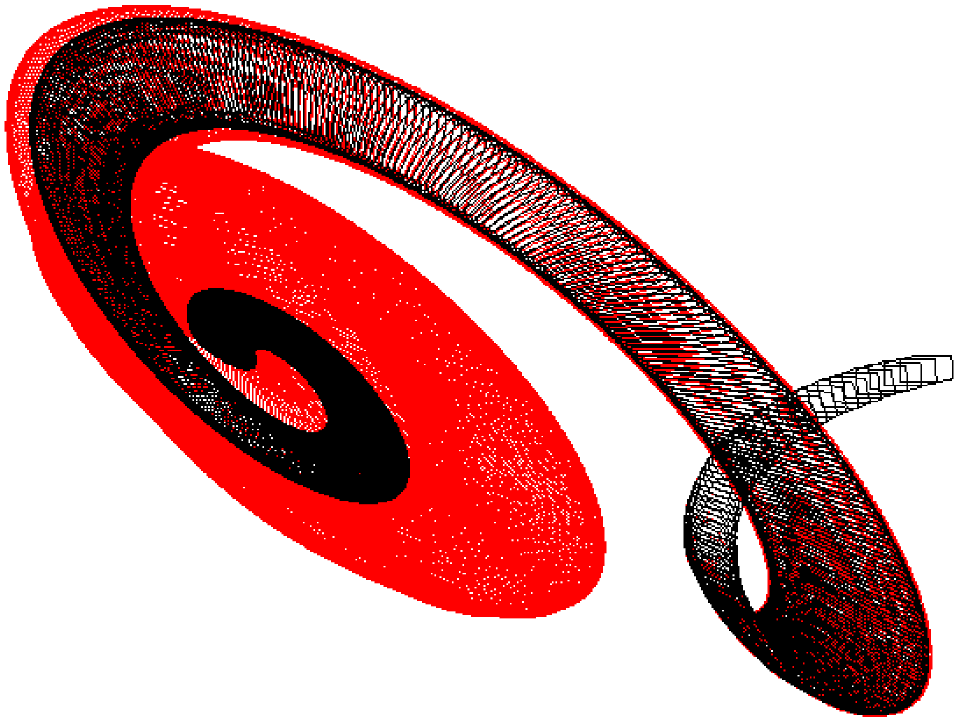
- **Fight complexity by overapproximation**
- **Overapproximated Sequence**

$$\hat{\Omega}_{k+1} \quad = \quad Approx(e^{A\delta}\hat{\Omega}_k \oplus V)$$

  - accumulation of approximations $\rightarrow$ Wrapping Effect
  - exponential increase in approximation error!

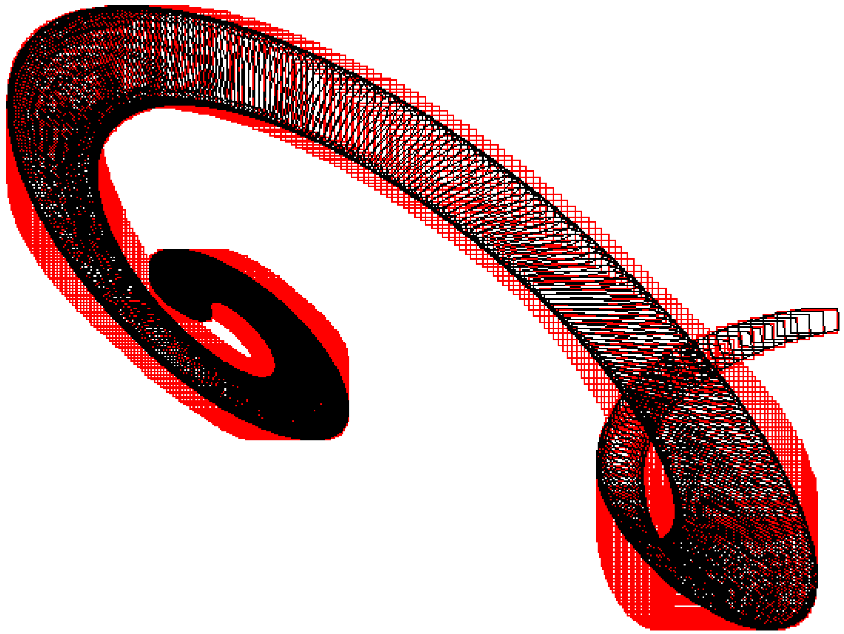
Wrapping Effect

- **Error Propagation in Conventional Algorithm:**



Wrapping Effect-Free Algorithm

- **Computing the sum of Sequences instead of a sequence of sums** [Girard, LeGuernic, Maler, 2006]



Outline

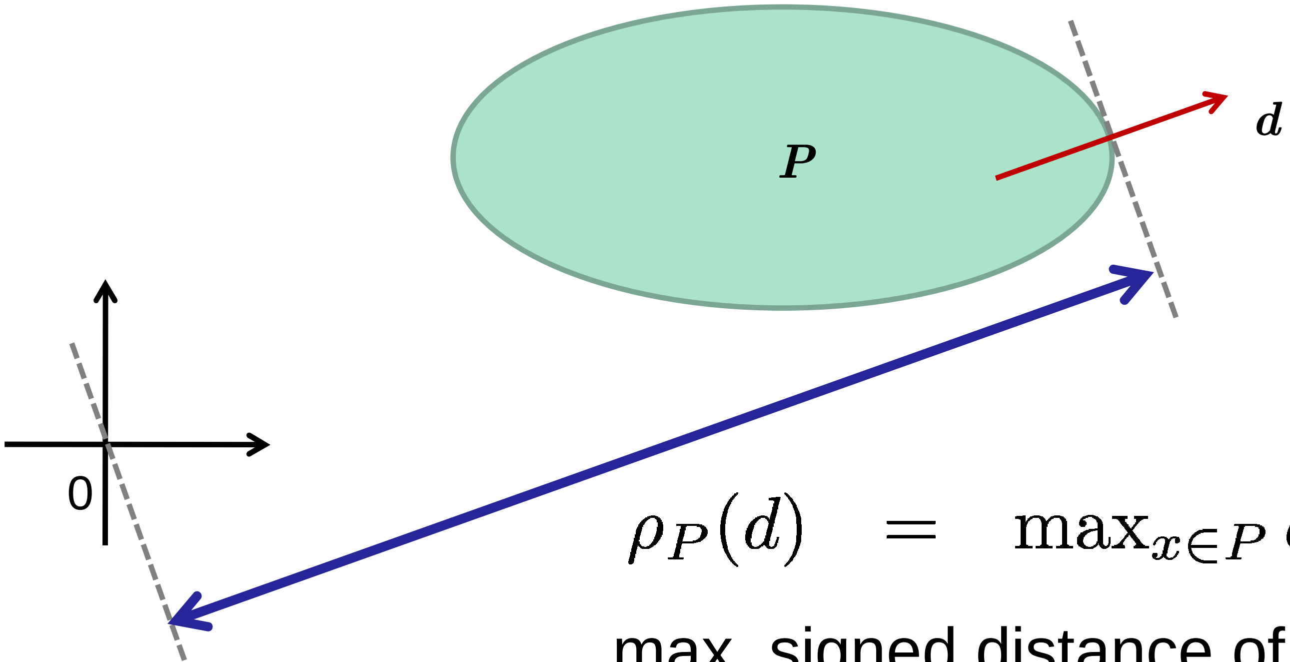
I. Hybrid Automata and Reachability

II. Linear Hybrid Automata

III. Piecewise Affine Hybrid Systems

IV. Support Functions

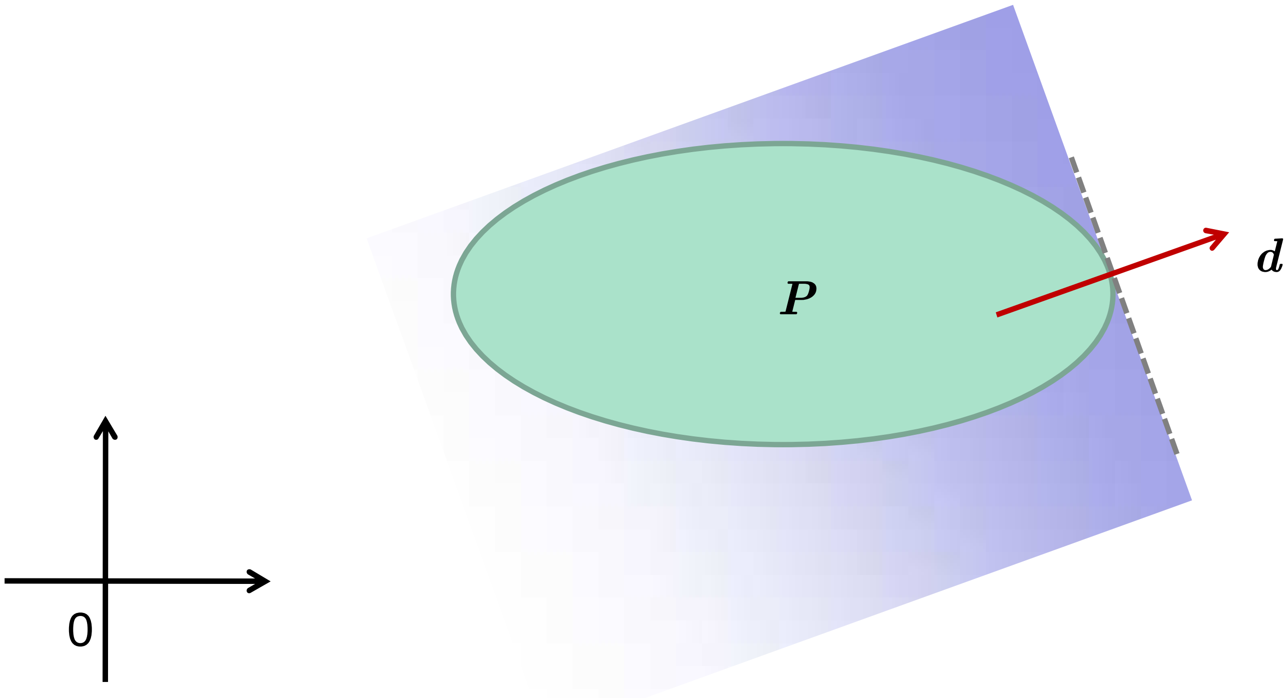
Support Functions



$$\rho_P(d) \quad = \quad \max_{x \in P} d^T x$$

max. signed distance of P to origin projected in direction d

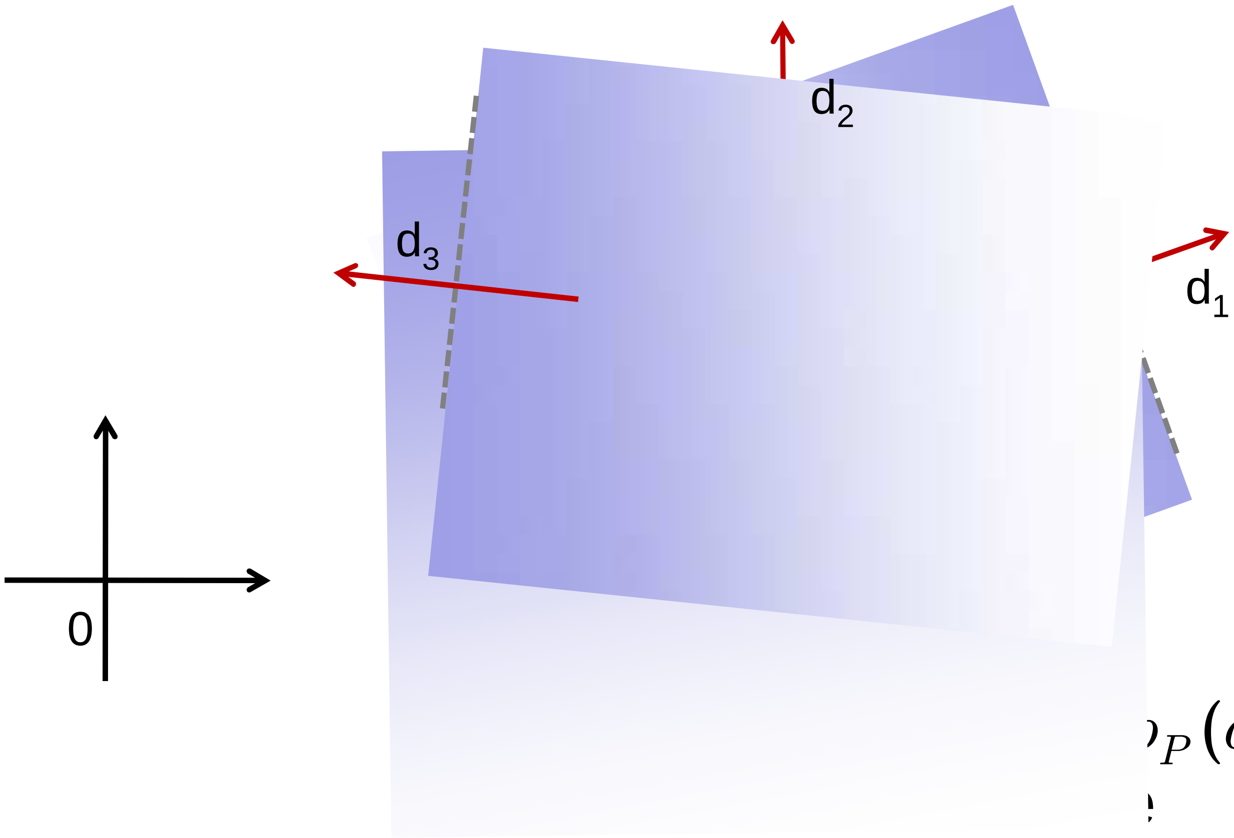
Support Functions



If we know the value of $\rho_P(d)$, we know P is in the halfspace

$$\{x \mid d^T x \leq \rho_P(d)\}$$

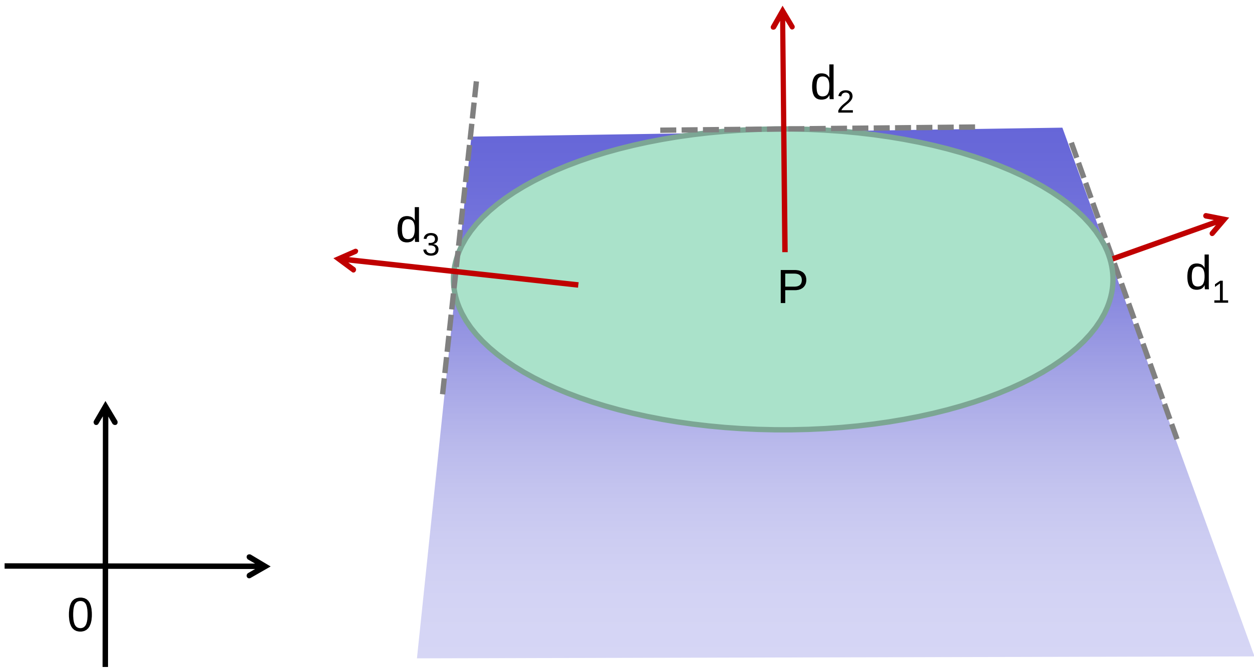
Support Functions



$\rho_P(d_2)$,... we

intersection of the halfspaces

Support Functions



If we know $\rho_P(d_\mathbf{1})$, $\rho_P(d_\mathbf{2})$,… we know P is inside the intersection of the halfspaces

= outer polyhedral approx.

Computing with Support Functions

- **Many set operations are simple operations on support functions**
  - Affine Transform: $\rho_{AP}(d) = \rho_P(A^T d)$
  - Minkowski sum: $\rho_{P \oplus Q}(d) = \rho_P(d) + \rho_Q(d)$
  - Convex hull: $\rho_{chull(P,Q)}(d) = \max(\rho_P(d), \rho_Q(d))$

- **Problems:**
  - Containment: use outer/inner polyhedral approx.
  - Intersection: approx. intersection with halfspace cheap, with polyhedron = multivariable optim. problem

Comparison of Set Representations

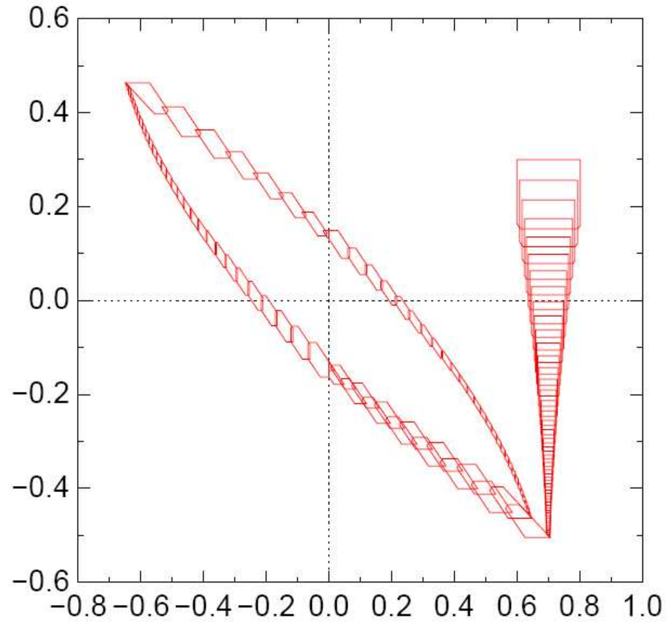
Operators	Polyhedra		Zonotopes	Support Functions
	Constraints	Vertices		
Affine transform	-	++	++	++
Minkowski sum	--	--	++	++
Intersection	++	--	--	+/-
Containment	+	--	?	+/(-)
Convex hull	--	+	--	++

Computing with Support Functions

- **If explicit set representation needed (display, simplification,…), sample the support function for given directions and use the outer polyhedral approximation.**
  - arbitrarily close if enough directions are used
- **Computing the support function of a polyhedron**
  - solve linear program (very cheap)

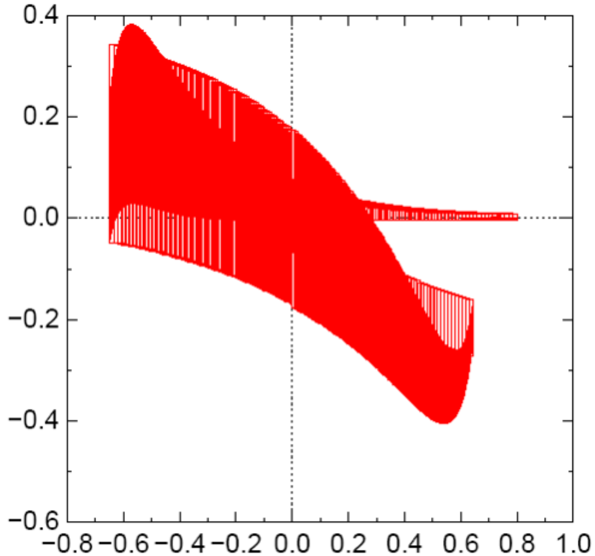
Filtered Switched Oscillator

- **Switched oscillator**
  - 2 state variables
  - similar to many circuits (Buck converters,…)
- **plus m^{th} order filter**
  - damps output signal
- **Piecewise affine dynamics**
  - 4 discrete states
  - total 2+m continuous state variables

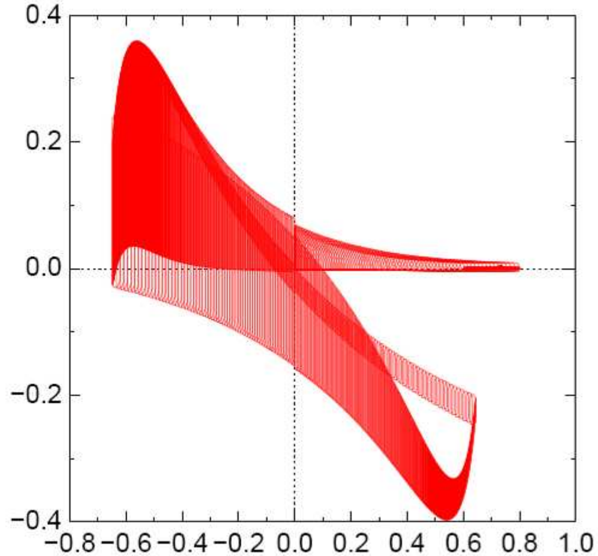


Filtered Switched Oscillator

- **2^{nd} order oscillator + 8^{th} order filter**
  - 10 state variables



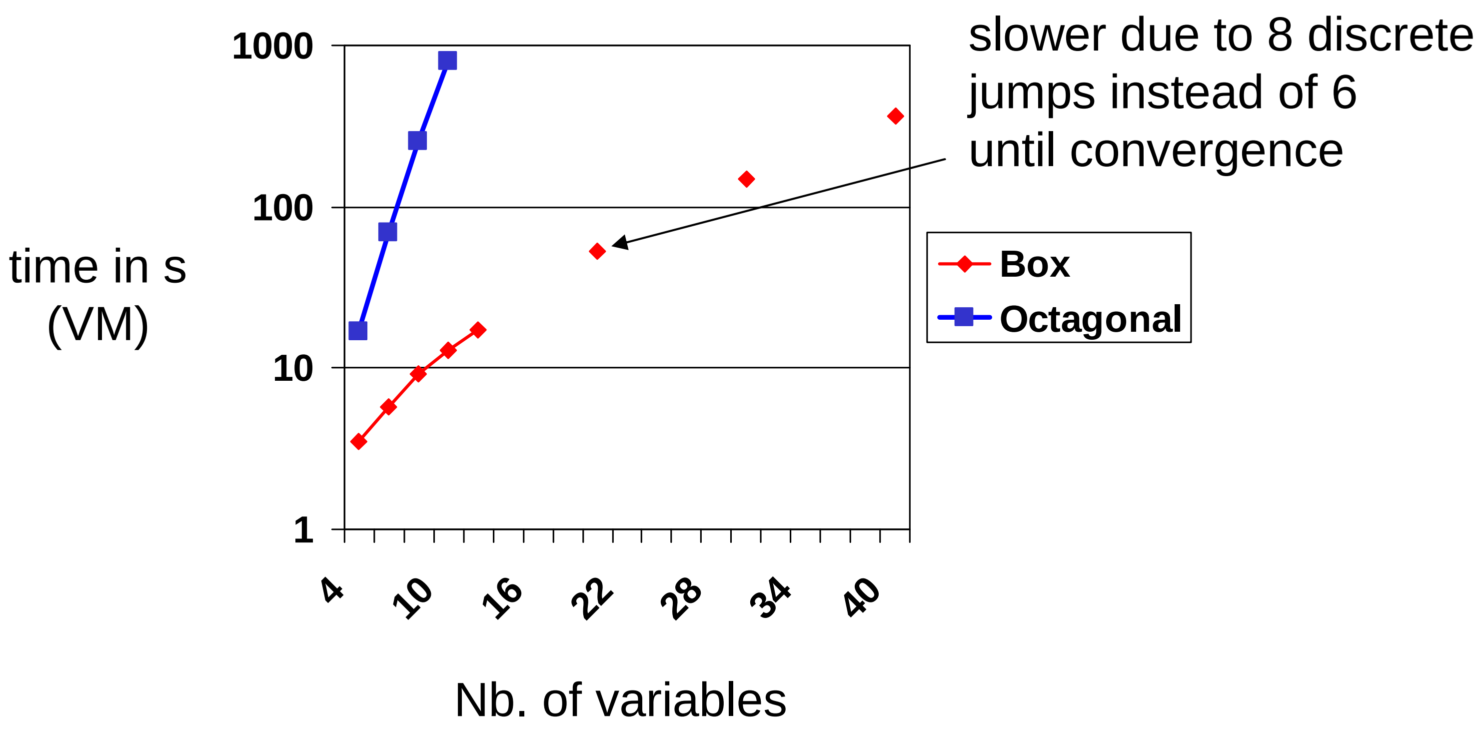
2*n box constraints
(axis directions)



$2{*}n^2$ octagonal constraints
($\pm x_i \pm x_j$)

Filtered Switched Oscillator

- **Tool Performance (on virtual machine)**



Bibliography

- **Hybrid Systems Theory**
  - Rajeev Alur, Costas Courcoubetis, Nicolas Halbwachs, Thomas A. Henzinger, Pei-Hsin Ho, Xavier Nicollin, Alfredo Olivero, Joseph Sifakis, and Sergio Yovine. The algorithmic analysis of hybrid systems. Theoretical Computer Science 138:3-34, 1995
  - Thomas A. Henzinger. The theory of hybrid automata. Proceedings of the 11th Annual Symposium on Logic in Computer Science (LICS), IEEE Computer Society Press, 1996, pp. 278-292
- **Linear Hybrid Automata**
  - Thomas A. Henzinger, Pei-Hsin Ho, and Howard Wong-Toi, HyTech: The next generation. RTSS'95
  - Goran Frehse. PHAVer: Algorithmic Verification of Hybrid Systems past HyTech. HSCC'05
  - Goran Frehse. Tools for the verification of linear hybrid automata models. In J. Lunze and F. Lamnabhi-Lagarrigue, editors, Handbook of Hybrid Systems Control. Cambridge University Press, 2009.

Bibliography

- **Affine Dynamics**
  - E. Asarin, O. Bournez, T. Dang, and O. Maler. Approximate Reachability Analysis of Piecewise-Linear Dynamical Systems. HSCC'00
  - A. Girard, C. Le Guernic, and O. Maler. Efficient computation of reachable sets of linear time-invariant systems with inputs. HSCC'06
- **Support Functions**
  - C. Le Guernic, A.Girard. Reachability analysis of hybrid systems using support functions. CAV'09
  - G. Frehse, R. Ray. Design Principles for an Extendable Verification Tool for Hybrid Systems. ADHS'09

Verification Tools for Hybrid Systems

- **HyTech: LHA**
  - http://embedded.eecs.berkeley.edu/research/hytech/
- **PHAVer: LHA + affine dynamics**
  - http://www-verimag.imag.fr/~frehse/
- **d/dt: affine dynamics + controller synthesis**
  - http://www-verimag.imag.fr/~tdang/Tool-ddt/ddt.html
- **Matisse Toolbox: zonotopes**
  - http://www.seas.upenn.edu/~agirard/Software/MATISSE/
- **HSOLVER: nonlinear systems**
  - http://hsolver.sourceforge.net/
- **and more…**