Reachability for Continuous and Hybrid Systems

Oded Maler

CNRS - VERIMAG
Grenoble, France

RP, September 2009

Preface

- This talk has two parts
- The first part presents work done in the "early days" of hybrid systems research, some 15 years ago
- It is about decidability and undecidability of some reachability problem for a simple type of hybrid automata
- This work is interesting and shows relations between computation, geometry and dynamics, but my current opinion is that this direction is not very applicable outside the paper industry
- The second part represents my current work in the domain
- We approximate reachable states of systems defined by linear and nonlinear differential equations
- I think this is a useful direction but I don't know what I will think about it in 15 years

Reachability Analysis of Dynamical Systems having Piecewise-Constant Derivatives

Eugene Asarin Oded Maler Amir Pnueli

CNRS - VERIMAG
Grenoble, France

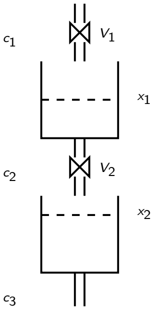
1993-1995

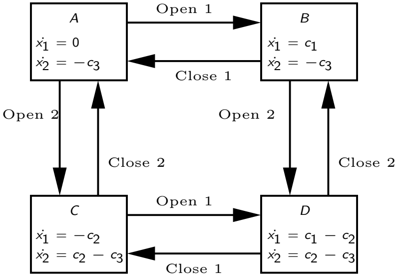
Outline of Talk

- Some generalities on “linear” hybrid automata and PCD systems
- Decidability of reachability problems in the plane
- Undecidability in dimension 3 and above by simulating pushdown stacks
- Going higher in the arithmetical hierarchy
- So what?

A Motivating Example: Buffer Networks

- Consider a network of containers/buffers for water/data
- Channels can be switched on and off
- When a channel is on, its flow rate is a constant
- Each combination of open/close valves leads to a different derivatives for the buffer levels, based on the difference between their in- and outflows



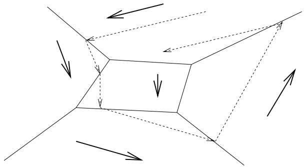


"Linear" Hybrid Automata and PCD Systems

- A sub-class of hybrid automata
- Can be viewed as piecewise-trivial dynamical systems: derivatives are constant in every control state (location) and the evolution is along a straight line
- Transition guards (switching surface) and invariants (staying conditions) are linear (hyperplanes, polytopes)
- Local continuous evolution needs no numerical analysis; Computing the effect of time passage amounts to quantifier elimination in linear algebra
- Investigated a lot by Henzinger et al. (HYTECH), currently supported by the tool PHAVER (G. Frehse)
- PCD (piecewise-constant derivative): a sub-class of linear hybrid automata closer in spirit to continuous dynamical systems

PCD (Piecewise-Constant Derivatives) Systems

- Dynamical System: $\mathcal{H} = (X, f)$, $X = \mathbb{R}^d$
- $f : X \to X$ defines differential equation $\frac{d^+\mathbf{x}}{dt} = f(\mathbf{x})$
- A trajectory of $\mathcal{H}$ starting at $\mathbf{x}_0 \in X$ is $\xi : \mathbb{R}_+ \to X$ s.t.
  - $\xi(0) = \mathbf{x}_0$
  - $f(\xi(t))$ is defined for every t and is equal to the right derivative of $\xi(t)$
- PCD: X is partitioned into a final number of polyhedra (regions) and f is constant in each region
- Trajectories are thus broken lines



PCDs are Effective

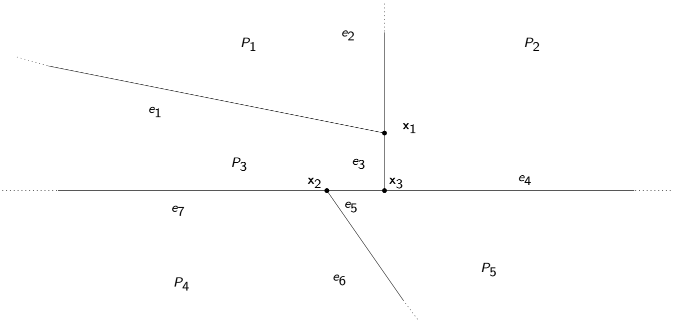
- A description of a PCD system: $\{(P_1, \mathbf{c}_1), \ldots, (P_n, \mathbf{c}_n)\}$
- each P_i is a convex polyhedron (interesection of linear inequalities) and $\mathbf{c}_i$ is its corresponding derivative (slope)
- Effectiveness: given a PCD description and a rational point $\mathbf{x} = \xi(0)$
- There exists $\epsilon > 0$ s.t. we can compute precisely $\mathbf{x}' = \xi(\Delta)$ for every Δ, $0 < \Delta t < \epsilon$; $x' = x + c \cdot \Delta$
- Unlike arbitrary dynamical systems where you can only approximate

Decision Problems for PCD

- Point-to-point reachability $\mathbf{Reach}(\mathcal{H}, \mathbf{x}, \mathbf{x}')$:
- Given: a PCD $\mathcal{H}$ and $\mathbf{x}, \mathbf{x}' \in X$,
- Are there a trajectory ξ and $t \geq 0$ such that $\xi(0) = \mathbf{x}$ and $\xi(t) = \mathbf{x}'$?
- Region-to-region reachability $\mathbf{R\text{-}Reach}(\mathcal{H}, P, P')$:
- Given: a PCD $\mathcal{H}$ and two polyhedral sets $P, P' \subseteq X$
- Are there two points $\mathbf{x} \in P$ and $\mathbf{x}' \in P'$ such that $\mathbf{Reach}(\mathcal{H}, \mathbf{x}, \mathbf{x}')$?

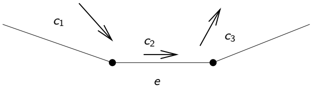
PCDs on the Plane

- Polyhedral partition of the plane into polygons/regions (P)
- Induced boundary elements: edges (e) and vertices (x)
- A kind of abstract finite alphabet to describe qualitative behaviors as sequences of regions or edges

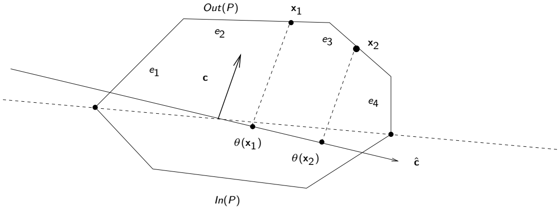


Orientation and Ordering of Boundaries

- Edges (and vertices) can be classified as entry and exit according to the relation between the slope c and the the vector e which defines the inequality
- Edge e below is exit for c_1 and entry for c_3

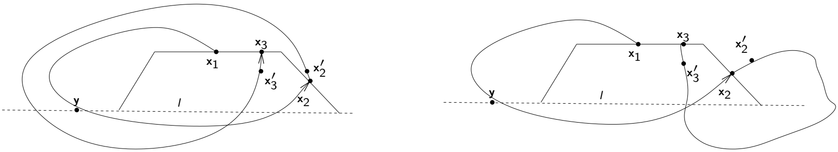


- The whole boundary of a region can be decomposed into two connected sets, entry $In(P)$ and exit $Out(p)$
- A linear order can be imposed on each of them:



A Fundamental Property of Planar Systems

- Let ξ be any trajectory that intersects $Out(P)$ in three consecutive points, $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$. Then: $\mathbf{x}_1 \preceq \mathbf{x}_2$ implies $\mathbf{x}_2 \preceq \mathbf{x}_3$

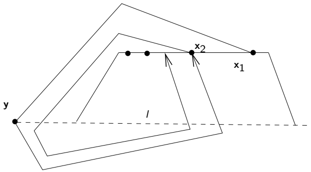


- The figure shows why it cannot be otherwise as the trajectory must intersect itself
- Jordan's theorem, not true in 3 dimensions

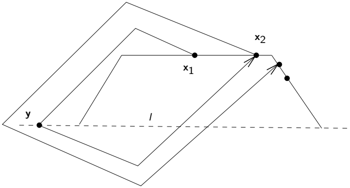
Spirals

- Consequently all repetitive behaviors are spirals

Contracting:



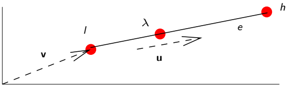
Expanding:



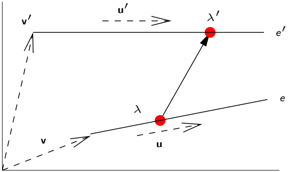
- The sequences of intersections with an edge is monotonic and you cannot return to an edge you have "abandoned"
- Since there are finitely many edges we can conclude:
- For every trajectory, the sequence of edges it crosses is ultimately-periodic: $e_1, \ldots, e_i, (e_{i+1}, \ldots, e_{i+j})^\omega$

Representation (Parametrization)

- A representation scheme for an edge e is a pair of vectors $\mathbf{v}, \mathbf{u}$ and an interval $[l, h]$ such that $e = \{\mathbf{v} + \lambda\mathbf{u} : \lambda \in [l, h]\}$



- Consider and entry edge e with $(\mathbf{u}, \mathbf{v})$ representation and exit edge e' with $(\mathbf{u}', \mathbf{v}')$ representation
- The corresponding successor function is defined as $f_{e,e'}(\lambda) = \lambda'$ iff by entering P at $\mathbf{x} = (e, \lambda)$, you exit as $\mathbf{x}' = (e', \lambda')$



Successor Function is Linear

- Successor function is well-defined, computable and linear: $\lambda' = A_{e,e'}\lambda + B_{e,e'}$ where

$$A_{e,e'} = \frac{\mathbf{c} \cdot \mathbf{a}}{\mathbf{c} \cdot \mathbf{a}'} \quad \text{and} \quad B_{e,e'} = \frac{\hat{\mathbf{c}} \cdot (\mathbf{v} - \mathbf{v}')}{\mathbf{c} \cdot \mathbf{a}'}$$

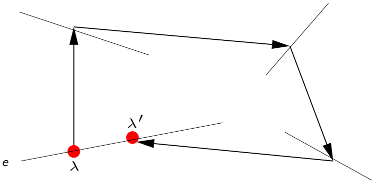
- Here c is the slope and a and a' are the normals to e and e'
- (Some basic linear algebra, quantifier elimination...)
- Predecessor:

$$\lambda = \frac{\lambda' - B_{e,e'}}{A_{e,e'}}$$

- Moreover: if $e \in In(P)$ and $e' \in Out(P)$ then $A_{e,e'} > 0$

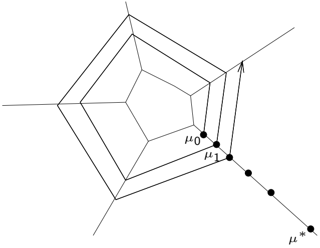
Signature Successor Function

- A cyclic signature: a sequence $\sigma = e_1, \ldots, e_k$ of edges s.t. $e_1 = e_k$



- The function f_σ from e_1 to itself represents the effect on a point going through a cycle (Poincare map)
- In our case it is linear $f_\sigma(\lambda) = A_\sigma \lambda + B_\sigma$ (composition of linear partial functions)
- $A_\sigma = A_{e_1,e_2} \cdot A_{e_2,e_3} \ldots A_{e_{k-1},e_k}$
- $B_\sigma = (\cdots((B_{e_1,e_2} \cdot A_{e_2,e_3} + B_{e_2,e_3}) \cdot A_{e_3,e_4} + B_{e_3,e_4}) \cdots) \cdot A_{e_{k-1},e_k} + B_{e_{k-1},e_k}$

Intersections of a Spiral and an Edge



- $\mu_{i+1} = A_\sigma \cdot \mu_i + B_\sigma$
- $\mu_n = \begin{cases} \mu_0 + B_\sigma \cdot n & \text{if } A_\sigma = 1 \\ \mu_0 \cdot A_\sigma^n + B_\sigma \cdot \dfrac{A_\sigma^n - 1}{A_\sigma - 1} & \text{otherwise} \end{cases}$
- We can compute $\mu^* = lim_{n \to \infty} \mu_n$

The Limit of the Sequence

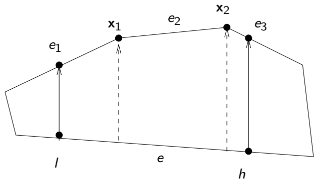
Case	*Limit*
$A_\sigma = 1, B_\sigma = 0$	μ_0
$A_\sigma = 1, \lvert B_\sigma \rvert > 0$	∞
$A_\sigma = 1, \lvert B_\sigma \rvert < 0$	$-\infty$
$A_\sigma < 1$	$\dfrac{B_\sigma}{1 - A_\sigma}$
$A_\sigma > 1, \mu_0 = \frac{B_\sigma}{1-A_\sigma}$	μ_0
$A_\sigma > 1, \mu_0 > \frac{B_\sigma}{1-A_\sigma}$	∞
$A_\sigma > 1, \mu_0 < \frac{B_\sigma}{1-A_\sigma}$	$-\infty$

Main Positive Result

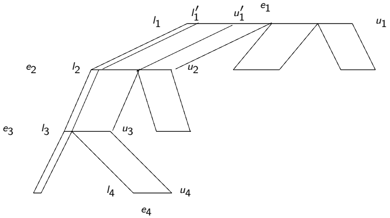
- An algorithm for deciding $\mathbf{Reach}(\mathcal{H}, \mathbf{x}, \mathbf{x}')$:
- Start "simulating" forward from $\mathbf{x}$
- When you encounter a cycle, compute its limit points on all edges and determine whether it is the ultimate cycle (limits on each edge stays inside edge range)
- If not, continue simulating until you leave it (in a finite number of iterations)
- If it is the ultimate cycle, and $\mathbf{x}'$ is beyond the limit, the answer is "no"
- If $\mathbf{x}'$ is before the limit then continue simulation until you reach $\mathbf{x}'$ ("yes") or bypass it ("no")

Region-to-Region Reachability (Sketch)

- Can be reduced to edge-to-edge reachability
- An entry edge interval splits into finitely many exits edges



- Can build a successor tree and compute a limit along each branch

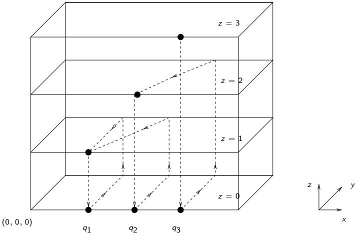


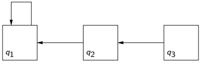
Can we go to Higher Dimensions?

- One one hand: calculating successors can be generalized to higher dimensions (more book-keeping though)
- On the other: no Jordan theorem so trajectories are not necessary ultimately-periodic (Chaos et co.)
- We show undecidability for 3 dimensions by showing that PCDs can simulate any TM (2PDA) and hence deciding reachability for PCDs solves the halting problem
- Interesting "model of computation"

Simulation of Finite-State Automata

- Every finite deterministic automaton can be simulated by a 3-dimensional PCD system





Region	Defining conditions	$\mathbf{c} = (\dot{x}, \dot{y}, \dot{z})$
F	$(z = 0) \wedge (y < 1)$	$(0, 1, 0)$
U_{ij}	$(x = i) \wedge (y = 1) \wedge (z < j)$	$(0, 0, 1)$
B_{ij}	$(z = j) \wedge (x + (j - i)y = j) \wedge (y > 0)$	$(j - i, -1, 0)$
D	$(z > 0) \wedge (y = 0)$	$(0, 0, -1)$

- Regions U_{ij} and B_{ij} are defined for every i, j such that $\delta(q_i) = q_j$

Push-down Automata (PDA)

- Pushdown stack: an element of $\Sigma^* 0^\omega$.
- Two operations:

$$\text{PUSH}: \Sigma \times \Sigma^\omega \to \Sigma^\omega \qquad \text{POP}: \Sigma^\omega \to \Sigma \times \Sigma^\omega$$

$$\text{PUSH}(v, S) = v \cdot S \qquad \text{POP}(v \cdot S) = (v, S)$$

- PDA: an infinite transition system $\mathcal{A} = (Q \times \Sigma^* 0^\omega, \delta)$
- Q is finite and δ is defined using a finite collection of statements of one of the following forms:

$$\begin{array}{lllll} q_i: & S := \text{PUSH}(v, S); & \qquad & q_i: & (v, S) := \text{POP}(S); \\ & \text{GOTO } q_j & & & \text{IF } v = 0 \text{ GOTO } q_{i_0}; \\ & & & & \dots \\ & & & & \text{IF } v = k-1 \text{ GOTO } q_{i_{k-1}}; \end{array}$$

Encoding Stacks into $[0, 1]$

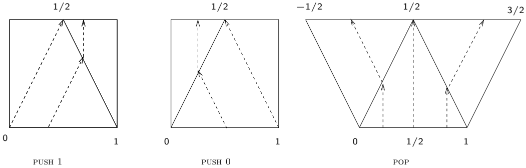
- Contents of a stack $S = s_1 s_2 \dots$ where s_1 is the top of the stack
- Enconding using k-ary representation $r : \Sigma^\omega \to [0, 1]$

$$r(S) = \sum_{i=1}^{\infty} s_i k^{-i}$$

- Stack operations have arithmetic counterparts:

$$\begin{array}{rlll} S' = & \textsc{push}(v, S) & \text{iff} & r(S') = (r(S) + v)/k \\ (S', v) = & \textsc{pop}(S) & \text{iff} & r(S') = kr(S) - v \end{array}$$

Building Blocks for the Simulation, $k = 2$ and $\Sigma = \{0, 1\}$

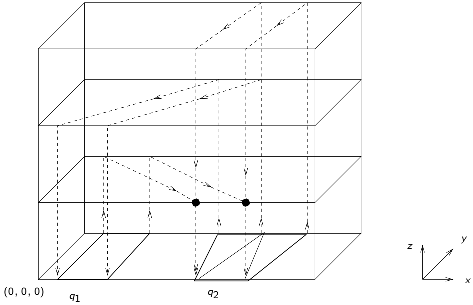


- A trajectory starting at $\mathbf{x} = (x, 0)$, $x \in [0, 1]$ and ending at $\mathbf{x}' = (x', 1)$ satisfies:
- $x' = (x + 1)/2$ (PUSH 1), $x' = x/2$ (PUSH 0) and $x' = 2x - 1/2$ (POP)
- In other words, $x = r(S)$ at the "input port" $(y = 0)$ of an element, then $x' = r(S')$ at the "output port" $(y = 1)$ where S' is the operation outcome.
- The POP element has two output ports which are selected according to the value of the top element popped

Simulation of PDAs by PCDs

- Put the appropriate element for each state and connect via "bands" that "carry" the stack value
- A PCD for the PDA defined by:

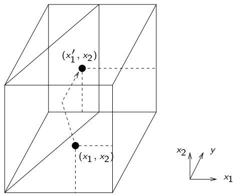
$q_1 : S :=\text{PUSH}(1, S);\ \text{GOTO}\ q_2;$
$q_2 : (v, S) :=\text{POP}(S);$ If $v = 1$ THEN GOTO q_2 ELSE GOTO q_1



- Every PDA can be simulated by a 3-dimensional PCD system

Simulating 2PDAs

- Automata with 2 push-down stacks can simulate Turing machines
- We can represent the configuration of two stacks by a point in $[0,1]^2$ and build the corresponding gadgets, e.g. $\text{PUSH}(S_1, 0)$



- Hence a straightforward realization of 2PDA in 4 dimensions
- With some considerable effort we can squeeze everything into 3 dimensions and conclude:
- The reachability problem for PCD systems in 3 dimensions is undecidable

Theoreticians go Wild

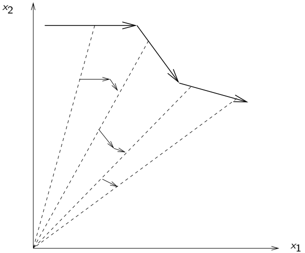
- Arithmetical hierarchy: the classes $\Sigma_1, \Sigma_2, \ldots$ and $\Pi_1, \Pi_2, \ldots$ of sets of integers defined inductively:
- Σ_1 consists of sets $P \subseteq \boldsymbol{N}$ such that there is a Turing machine that halts on an input n iff $n \in P$
- The class Π_i consists of all the sets P such that $\overline{P} \in \Sigma_i$
- Σ_{i+1} is the class of all sets P defined as $P = \{n : \exists m \langle m, n \rangle \in P'\}$ for some $P' \in \Pi_i$, where $\langle\rangle$ is some computable pairing function
- The arithmetical hierarchy is infinite, satisfying the strict inclusions $\Pi_i \subset \Sigma_{i+1}$ and $\Sigma_i \subset \Pi_{i+1}$
- We show (with the help of Zeno paradox) how all the arithmetical hierarchy can be realized by PCDs

Recognition by PCDs

- PCD recognizer: $\widehat{\mathcal{H}} = (\mathbb{R}^d, f, I, r, \mathbf{x}^{\mathrm{A}}, \mathbf{x}^{\mathrm{R}})$, $\mathcal{H} = (\mathbb{R}^d, f)$ is a PCD
- $I = [0,1] \times \{0\}^{d-1}$ is a one-dimensional subset of X (the "input port")
- $r : \boldsymbol{N} \to [0,1] \cap \mathcal{Q}$ is a recursive injective coding function
- $\mathbf{x}^{\mathrm{A}}, \mathbf{x}^{\mathrm{R}} \in \mathbb{R}^d - I$ are two distinct points (accepting and rejecting states)
- We assume that $f(\mathbf{x}^{\mathrm{A}}) = f(\mathbf{x}^{\mathrm{R}}) = 0$
- $\widehat{\mathcal{H}}$ semi-recognizes $P \subseteq \mathbb{N}$ iff for every n, the trajectory starting at $(r(n), 0, \ldots, 0)$ can continue forever and it eventually reaches $\mathbf{x}^{\mathrm{A}}$ iff $n \in P$
- We say that $\hat{\mathcal{H}}$ (fully) recognizes P when, in addition, this trajectory reaches $\mathbf{x}^{\mathrm{R}}$ iff $n \notin P$
- Previous result: every Σ_1 set P is semi-recognized by some 3-dimensional bounded PCD

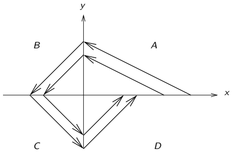
Principal Lammata

- From a PCD that semi-recognizes P one can construct a (higher-dimensional) PCD that recognizes P
- From a PCD that recognizes P one can construct:
  1. a PCD that semi-recognizes $\{x : \exists y \langle x, y\rangle \in P\}$
  2. a PCD that recognizes $\overline{P}$.
- The last two are relatively-easy and trivial (respectively)
- The main idea of the first:

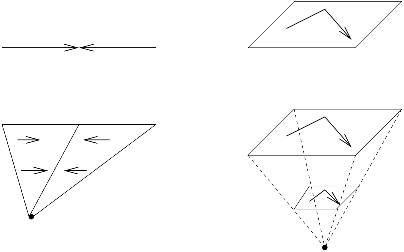


Gadgets used in the Construction

- Division by 2:



- Projectivisation:



- Corollary: PCDs can realize the whole arithmetical hierarchy

Credits and Follow-ups

- Decidability : OM and A. Pnueli, Reachability Analysis of Planar Multi-Linear Systems, 1993
- Generalized by Asarin, Pace, Schneider and Yovine to planar differential inclusions (and implemented)
- Undecidability: E. Asarin and OM, On some Relations between Dynamical Systems and Transition Systems, 1994
- Numerous papers on decidability boundaries for linear hybrid automata (Henzinger et al)
- Some small open problems remain, e.g. M. Mahfoudh, B. Krogh and OM, On Control with Bounded Computational Resources, 2002
- Higher undecidability: E. Asarin and OM, Achilles and the Tortoise Climbing Up the Arithmetical Hierarchy, 1995
- Studied extensively by O. Bournez

So What?

- Beyond the nice intellectual exercise (and a warm-up for those whose geometry and linear algebra are, at best, rusty) the results are rather disappointing
- Even for these systems, whose continuous dynamics is trivial we cannot answer anything
- How will we cope with "real" dynamics?
- We are asking the wrong questions, inspired by our discrete verification background
- In the continuous world having precise/exact answers is an oxymoron
- We should ask weaker, approximate questions on stronger systems with real differential equations

Computing Reachable States for Nonlinear Biological Models

Thao Dang   Colas Le Guernic   Oded Maler

CNRS - VERIMAG
Grenoble, France

2009

Summary

- We propose a computer-aided methodology to help analyzing certain biological models
- Domain of applicability: **biochemical reactions** modeled as **differential equations**. State variables denote **concentrations**
- We propose **reachability computation**, a kind of **set-based simulation**, that may replace uncountably-many simulations
- The continuous analogue of **algorithmic verification** (model-checking), emerged from more than a decade of research on **hybrid systems**
- Since this is not part of the local culture, we first introduce the domain and only later move to the contribution of this paper

Outline

- **Under-determined** dynamical models and their biological relevance
- **Continuous dynamical systems** and abstract reahcability
- **Effective representation** of sets and concrete algorithms for **linear** systems
- Treating **nonlinear** systems via **hybridization**
- **Dynamic hybridization**: idea and preliminary results
- Conclusions

Dynamical Models with Nondeterminism

- Dynamical system: state space X and a rule $x' = f(x, v)$
- The **next state** as a function of the **current state** and some **external influence** (or unknown parameters) $v \in V$
- In discrete domains: a transition system with input (alphabet)
- System becomes nondeterministic if input is projected away
- Given initial state, many possible evolutions ("runs")
- **Simulation**: picking **one** input and generating **one** behavior
- **Symbolic verification**: magically computing **all** runs in parallel
- **Reachability computation**: adapting these ideas to systems defined by **differential equations** or **hybrid automata** (differential equations with mode switching)

Why Bother?

- Differential models of biochemical reactions are **very** imprecise for many reasons:
- They are obtained by measuring populations, not individuals
- Kinetic parameters are based on isolated experiments not always under same conditions
- Etc.
- It is nice to match an experimentally-observed behavior by a deterministic model, but can we do better?
- After all, biological systems are supposed to be **robust** under **variations** in environmental conditions and parameters
- Showing that **all** trajectories corresponding to a **range** of parameters exhibit the same **qualitative behavior** is much stronger

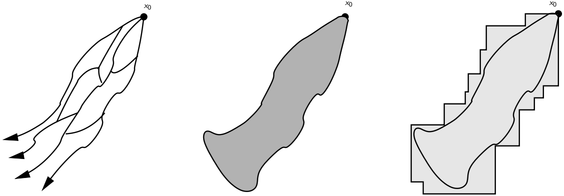
Preliminary Definitions and Notations

- A time domain $T = \mathbb{R}_+$, state space $X \subseteq \mathbb{R}^n$, input space $V \subseteq \mathbb{R}^m$
- **Trajectory**: partial function $\xi : T \to X$, **Input signal**: $\zeta : T \to V$ both defined over an interval $[0, t] \subset T$
- A continuous dynamical system $S = (X, V, f)$
- Trajectory ξ with endpoints x and x' is the **response** of S to input signal ζ if
- ξ is the solution of $\dot{x} = f(x, v)$ for initial condition x and $v(\cdot) = \zeta$, denoted by $x \xrightarrow{\zeta/\xi} x'$
- $R(x, \zeta, t) = \{x'\}$ denote the fact that x' is reachable from x by ζ within t time, that is, $x \xrightarrow{\zeta/\xi} x'$ and $|\zeta| = |\xi| = t$

Reachability

- $R(x, \zeta, t) = \{x'\}$ speaks of **one** initial state, **one** input signal and **one** time instant
- Generalizing to a **set** X_0 of initial states, to **all** time instants in an interval $I = [0, t]$ and **all** admissible input signals:

$$R_I(X_0) = \bigcup_{x \in X_0} \bigcup_{t \in I} \bigcup_{\zeta} R(x, \zeta, t)$$



- Depth-first vs. breadth-first

$$\bigcup_{\zeta} \bigcup_{t \in I} R(x, \zeta, t) = \bigcup_{t \in I} \bigcup_{\zeta} R(x, \zeta, t)$$

Abstract Reachability Algorithm

- The reachability operator satisfies the semigroup property:

$$R_{[0,t_1+t_2]}(X_0) = R_{[0,t_2]}(R_{[0,t_1]}(X_0))$$

- We can choose a time step r and apply the following iterative algorithm:

Input: A set $X_0 \subset X$
Output: $Q = R_{[0,L]}(X_0)$

$P := Q := X_0$
repeat $i = 1, 2 \ldots$
  $P := R_{[0,r]}(P)$
  $Q := Q \cup P$
until $i = L/r$

- Remark: we look at bounded time horizon and do not mind about reaching a fixpoint

From Abstract to Concrete Algorithms

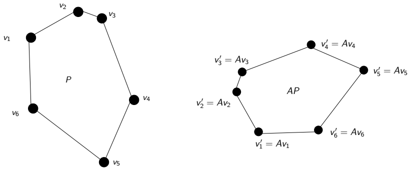
- The algorithm performs operations on **subsets** of $\mathbb{R}^n$ which, mathematically speaking, can be weird objects
- Like any **computational** geometry we restrict ourselves to classes of subsets (boxes, polytopes, ellipsoids, zonotopes) having nice properties:
- **Finite** syntactic representation
- Effective decision procedure for membership
- **Closure** (or approximate closure) under the reachability operator
- In this talk we use **convex polytopes** and their finite unions

Convex Polytopes

- **Halfspace**: all points x satisfying a linear inequality $a \cdot x \leq b$
- **Convex polyhedron**: intersection of finitely many halfspaces; **Polytope**: bounded convex polyhedron
- **Convex combination** of a set of points $\{x_1, \ldots, x_l\}$ is any $x = \lambda_1 x_1 + \cdots + \lambda_l x_l$ such that $\sum_{i=1}^{l} \lambda_i = 1$
- The **convex hull** $conv(\tilde{P})$ of a set $\tilde{P}$ of points is the set of all convex combinations of elements in $\tilde{P}$
- Polytope representations:
  - **Vertices**: a polytope P admits a finite minimal set $\tilde{P}$ (vertices) such that $P = conv(\tilde{P})$.
  - **Inequalities**: a polytope P admits a canonical set of halfspaces/inequalities such that $P = \bigwedge_{i=1}^{k} a^i \cdot x \leq b^i$

Autonomous (Closed, Deterministic) Linear Systems

- Systems defined by linear differential equations of the form $\dot{x} = Ax$ where A is a matrix are the most well-studied
- There is a standard technique to fix a time step r and work in discrete time, a **recurrence equation** of the form $x_{i+1} = Ax_i$
- The image of a set P by the linear transformation A is $AP = \{Ax : x \in P\}$ (one-step successors)
- It is easy to compute, for example, for polytopes represented by vertices:
  $P = conv(\{x_1, \ldots, x_l\}) \Rightarrow AP = conv(\{Ax_1, \ldots, Ax_l\})$



Algorithm 1: Discrete-Time Linear Reachability

- **Input**: A set $X_0 \subset X$ represented as $conv(\tilde{P}_0)$
- **Output**: $Q = R_{[0..L]}(X_0)$ represented as a list $\{conv(\tilde{P}_0), \ldots, conv(\tilde{P}_L)\}$

$$
\begin{array}{l}
P := Q := \tilde{P}_0 \\
\textbf{repeat } i = 1, 2 \ldots \\
\quad P := AP \\
\quad Q := Q \cup P \\
\textbf{until } i = L
\end{array}
$$

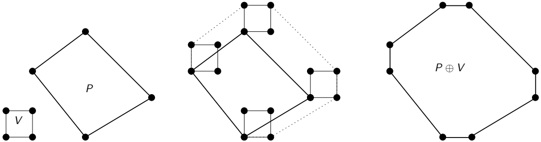
- Complexity assuming $|\tilde{P}_0| = m_0$ is $O(m_0 L M(n))$ where $M(n)$ is the complexity of matrix-vector multiplication in n dimensions: $\sim O(n^3)$
- Can be applied to other representations of objects closed under linear transformations

Linear Systems with Input

- Systems define by $x_{i+1} = Ax_i + v_i$ where the v_i's range over a bounded convex set V
- The one-step successor of P is defined as

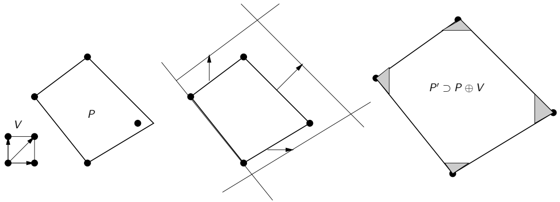
$$P' = \{Ax + v : x \in P, v \in V\} = AP \oplus V$$

- Minkowski sum $A \oplus B = \{a + b : a \in A \wedge b \in b\}$
- Same algorithm can be applied but the Minkowski sum increases the number of vertices in every step



Alternative: Pushing Facets

- Over-approximating the reachable set while keeping its complexity more or less fixed
- Assume P represented as intersection of halfspaces
- For each halfspace $H^i : a^i x \leq b^i$, let $v^i \in V$ be the input vector which pushes it in the "outermost" way
- Apply $Ax + Bv^i$ to H^i and the intersection of the pushed halfspaces over-approximates $AP \oplus V$



- The problem: over-approximation errors accumulate (the "wrapping effect")

Linear Reachability: State of the Art

- New algorithmics by C. Le Guernic and A. Girard
- Efficient computations: linear transformation applied to fixed number of points in each iteration
- No accumulation of over-approximation errors
- Initially used **zonotopes**, a class of sets closed under both linear operations and Minkowski sum; Can be applied to any "lazy" representation of the sequence of the computed sets
- Based on the observation that two consecutive sets

  $$\begin{array}{lcl} P_k & = & A^k P_0 \oplus A^{k-1} V \oplus A^{k-2} V \oplus \ldots \oplus V \\ P_{k+1} & = & A^{k+1} P_0 \oplus A^k V \oplus A^{k-1} V \oplus \ldots \oplus V \end{array}$$

  share a lot of terms
- Can compute within few minutes the reachable set after 1000 steps for linear systems with 200 (!) state variables

Linear Reachability: Some Credits

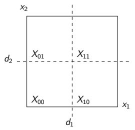
- Algorithmic analysis of hybrid systems started with tools like **Kronos** and **HyTech** for timed automata and "linear" hybrid automata: **HenzingerSifakisYovine** and **HenzingerHoWongtoi** - very simple continuous dynamics, summarized in **ACH^{+}95**
- Verifying differential equations: **Greenstreet96**
- Reachability for linear differential equations and hybrid systems: **ChutinanKrogh99**, **AsarinBournezDangMaler00** (polytopes) **KurzhanskiVaraiya00**, **BotchkarevTripakis00** (ellipsoids), **MitchellTomlin00** (level sets)
- Pushing faces and treating inputs: **DangMaler98**, **Varaiya98**
- Using zonotopes: **Girard05**
- New algorithmic scheme **Girard LeGuernic06-09**

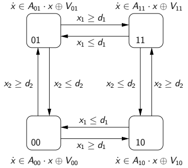
The Nonlinear Challenge

- Ok, bravo, but linear systems were studied to death by everybody. Real interesting models, biological included, are **nonlinear**
- What about systems of the form $x_{i+1} = f(x_i, u_i)$ or even $x_{i+1} = f(x_i)$ where f is an arbitrary continuous function, say a polynomial ?
- Convexity-preservation property of linear maps doesn't hold
- You can make small time steps, use a **local linear approximation** and bloat the obtained set to be safe
- This approach will either accumulate **large errors** or require expensive computation in **every step**

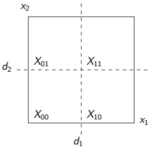
Hybridization: Asarin, Dang and Girard 2003

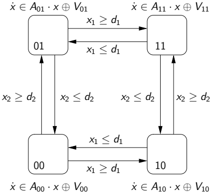
- Take a nonlinear system $x_{i+1} = f(x_i)$ and partition the state space into boxes (linearization domains)
- In each box X_q find a matrix A_q and a convex polytope V_q s.t. $f(x) \in A_q x \oplus V_q$ for every $x \in X_q$
- A_q is a **local linearization** of f with error bounded by V_q
- The new dynamics is $\quad x_{i+1} \in A_q x \oplus V_q \;$ iff $\; x \in X_q$
- A piecewise-(linear-with-input) systems, a restricted type of a hybrid automaton, which over-approximate f in terms of inclusion of trajectories





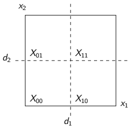
Hybridization (cont.)

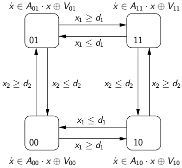




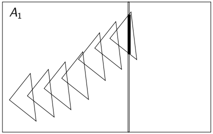
- In the hybrid automaton, x evolves according to the linear dynamics $A_q x \oplus V_q$ as long as it **remains** in X_q
- Reaching the **boundary** between X_q and $X_{q'}$, it takes a **transition** to q' and evolves according to $A_{q'} x \oplus V_{q'}$
- Linearization and error are computed only in the passage between blocks, **not** in every step
- Quality can be improved by making boxes smaller

Hybrid Reachability

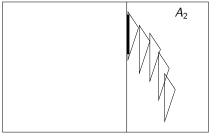




- Compute in one domain a sequences of sets using linear techniques until a set intersects with a boundary
- Take the intersection as initial set in next domain with the next linearization



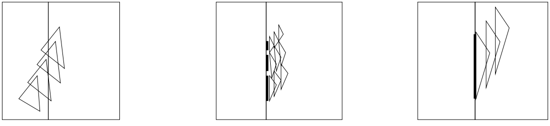
(a)



(b)

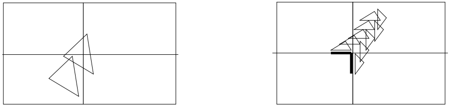
Between Theory and Practice

- First problem: intersection may be spread over many steps:



(a) (b) (c)

- Either **explosion** or union of intersections, **error** accumulation
- Major problem: a set may leave a box via **many facets**:

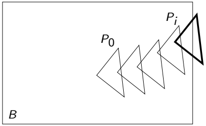


(a) (b)

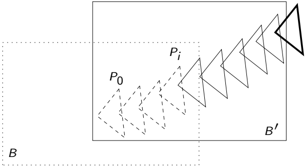
- Splitting is an **artifact** of the **fixed grid** imposed on the system
- Consequently, static hybridization is practically impossible beyond 3 dimensions

Our Contribution (at Last!)

- A **dynamic** hybridization scheme **not** based on a fixed grid
- In this scheme we do not need intersection **at all** and we allow the linearization domains to **overlap**
- When we leave a domain, we backtrack one step and define a new linearization domain around the previous set and continue with the new linearized dynamics from there



(a)

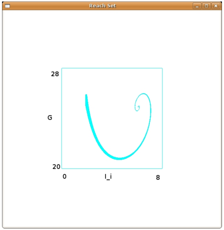


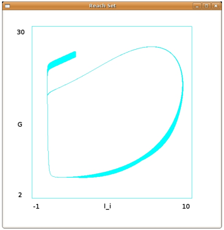
(b)

- And it works!

Example: E. Coli Lac Operon

$$
\begin{aligned}
\dot{R}_a &= \tau - \mu * R_a - k_2 R_a O_f + k_{-2}(\chi - O_f) - k_3 R_a I_i^2 + k_8 R_i G^2 \\
\dot{O}_f &= -k_2 r_a O_f + k_{-2}(\chi - O_f) \\
\dot{E} &= \nu k_4 O_f - k_7 E \\
\dot{M} &= \nu k_4 O_f - k_6 M \\
\dot{I}_i &= -2k_3 R_a I_i^2 + 2k_{-3} F_1 + k_5 I_r M - k_{-5} I_i M - k_9 I_i E \\
\dot{G} &= -2k_8 R_i G^2 + 2k_{-8} R_a + k_9 I_i E
\end{aligned}
$$





- We can also do a 9-dimensional highly-nonlinear **aging** model

Conclusions

- Disclaimer: we do **not** bring any new biological insight on any concrete system at this point
- Our goal is to develop **tools**, as **general-purpose** as possible, that can aid in the analysis of **many** non-trivial systems
- **Problem specificity** cannot be avoided of course: it will come up at the particular modeling and exploration phases
- Current version is a **prototype**:
  - Fixed-size boxes as linearizarization domains and other heuristics. Can be improved in efficiency and accuracy;
  - It is based on the old algorithmics for linear systems;
  - Improving all these aspects is on our immediate agenda
- We also explore alternative approaches for parameter synthesis based on simulation and sensitivity analysis **Donze et al09**
- Methodological aspects of the use of such tools in the **biological context** should be worked out

Thank You