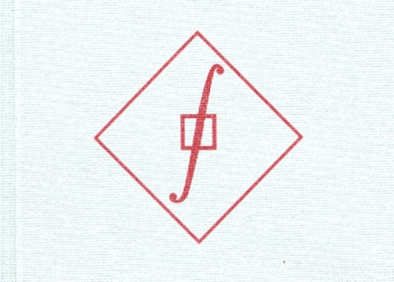
Continuous Systems Verification

Oded Maler

CNRS - VERIMAG
Grenoble, France

Amir Pnueli Memorial Symposium 2010



Introduction

- According to Manna and Pnueli, a verification framework has three ingredients:
- A **system model**: a formalism for describing the designed system (automata, transition systems, programs)
- A **specification language**: a formalism for describing the desired properties of the system. In other words a criterion for classifying event sequences as good or bad
- A **verification technique**: a method to show that (some/all) behaviors generated by the system are acceptable according to the specification

Introduction

- In this talk we focus on:
  - **System models** which are **continuous dynamical systems** defined by **differential equations**,
  - **algorithmic verification** against **simple** properties
- Initial motivation: **real-time**, **embedded**, **cyber-physical** and other buzzwordful systems where computers **control** a **physical** environment
- Additional collected motivations: new techniques in **applied mathematics**, verification of **analog circuits**, analyzing **biochemical reactions**
- We use the latter domain for motivation but the concepts and algorithms are rather generic

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**

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
- Appendix

Dynamical Models with Nondeterminism

- Dynamical system: state space X and a rule $x' = f(x, v)$
- The **next state** is 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 and external disturbances exhibit the same **qualitative behavior** is a much stronger potential contribution

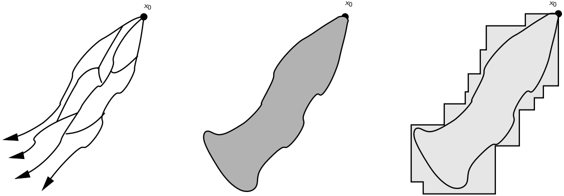
Preliminary Definitions and Notations

- 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 $[0, r] \subset T$
- 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 $I = [0, r]$ 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 a **bounded time horizon** and do not care about reaching a fixpoint

From Abstract to Concrete Algorithms

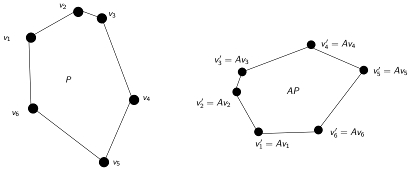
- The algorithm performs operations on **subsets** of $\mathbb{R}^n$ which, mathematically speaking, can be weird objects
- Like any **computational geometry** we restrict ourselves to classes of subsets (boxes, polytopes, ellipsoids, zonotopes) having nice properties:
- **Finite** syntactic representation
- 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 $\dot{x} = Ax$ for a matrix A 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 $\{conv(\tilde{P}_0), \ldots, conv(\tilde{P}_L)\}$

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

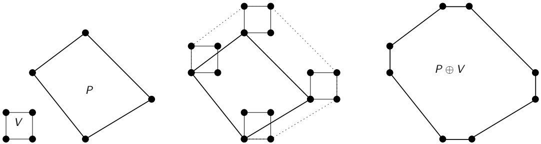
- Assuming $|\tilde{P}_0| = m_0$, the **complexity** of the algorithm is $O(m_0 L M(n))$ where $M(n)$ is the complexity of matrix-vector multiplication in n dimensions: $\sim O(n^3)$
- Can be applied to other representations of objects closed under linear transformations

Linear Systems with Input (Minkowski Sum Approach)

- 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

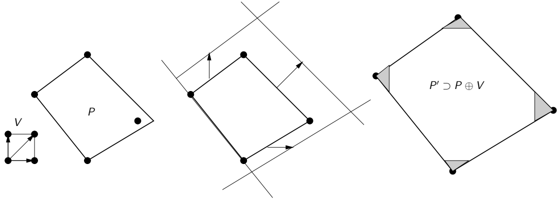
$$P' = \{Ax + v : x \in P, v \in V\} = AP \oplus V$$

- 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/facets in every step



Alternative: Face Lifting

- **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 enemy of the people is the **wrapping effect**: over-approximation errors accumulate every step

Linear State of the Art (Minkowski Approach)

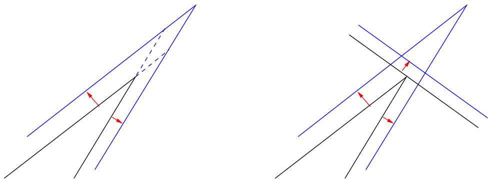
- New algorithmics by C. Le Guernic and A. Girard
- Efficient computations: linear transformation applied to a **fixed** number of points in each iteration
- **No accumulation** of over-approximation errors
- Initially used **zonotopes**, a class of sets closed under both linear operations and Minkowski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

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

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

Linear State of the Art (Optimization Approach)

- Recent result by T. Dang and R. Testylier
- Observation: over-approximation error on sharp corners can be significantly reduced by adding **redundant constraints**



- Moreover, the extra constraint can be added in the **right place** and **orientation**, after the over-approximating set intersects the bad set
- A kind of **dynamic approximation refinement**
- No need to move between constraint and vertex representations
- A prototype can easily handle 100 dimensions

Linear Reachability: Some Credits

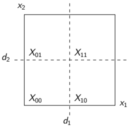
- Algorithmic analysis of hybrid systems started with tools like **Kronos** and **HyTech** for timed automata and "linear" hybrid automata: **HenzingerSifakisYovine**,**HenzingerHoWongtoi**
- Very simple continuous dynamics, summarized in **ACH^{+}95**
- Verifying differential equations: **Greenstreet96**
- 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 schemes **LeGuernic Girard06-09, DangTestylier10**

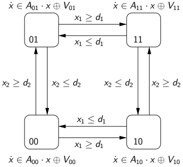
The Nonlinear Challenge

- 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 simply $x_{i+1} = f(x_i)$ where f is an arbitrary continuous function, say a polynomial ?
- Nonlinear maps do not preserve convexity
- 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 very expensive computation in **every step** of the main loop

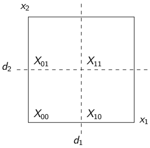
Hybridization: Asarin, Dang and Girard 2003

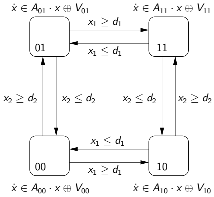
- Take a nonlinear system $x_{i+1} = f(x_i)$ and partition the state space into **linearization domains** (boxes, simplices)
- In each domain X_q find a matrix A_q and a convex polytope V_q s.t. $f(x) \in A_q x \oplus V_q$ for every $x \in X_q$
- A_q is a **local linearization** of f with error bounded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





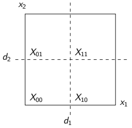
Hybridization (cont.)

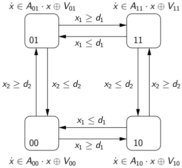




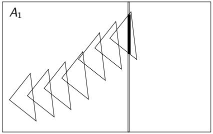
- In the hybrid automaton, x evolves according to the linear dynamics $A_q x \oplus V_q$ as long as it **remains** in X_q
- Reaching the **boundary** between X_q and $X_{q'}$, it takes a **transition** to q' and evolves according to $A_{q'} x \oplus V_{q'}$
- Linearization and error are recomputed only while **crossing** domain boundaries, **not** in every step
- Approximation quality can be tuned by controlling the size of linearization domains

Hybrid Reachability

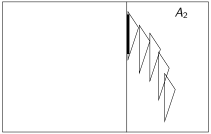




- Compute in one domain a sequences of sets using **linear techniques** until a set intersects with a boundary
- Take the intersection as **initial set** in the next domain and apply linear reachability with the corresponding linearization



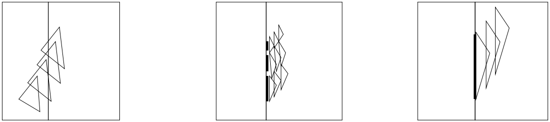
(a)



(b)

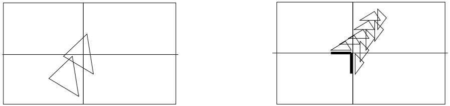
Between Theory and Practice

- 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**:

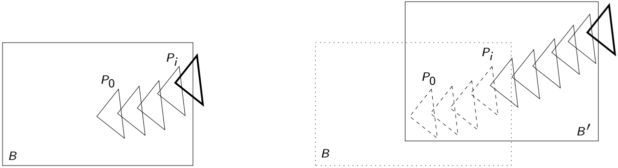


(a) (b)

- Consequently, **static** hybridization is practically impossible beyond 3 dimensions
- Set splitting is an **artifact** of the **fixed grid** that we violently imposed on Space

The Solution: Dynamic Hybridization

- 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

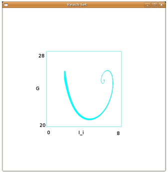


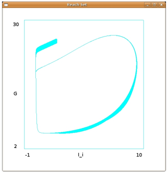
(a) (b)

- And it works!

Example: E. Coli Lac Operon

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





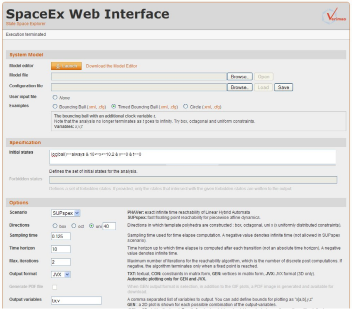
- We can also do a 9-dim highly-nonlinear **aging** model, and a model of an **angiogenesis** pathway (14-dim polynomial DE)

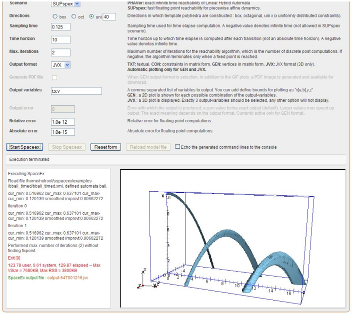
Conclusions

- 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
- Methodological aspects of the use of such tools in the **biological context** should be worked out
- Work in progress: optimizing the choice of size and orientation of the linearizarization domains
- Current version is still a **prototype**, based on the old algorithmics for linear systems, hence we are optimistic about going to even higher dimensions

Commercial I: SpaceEx

- Coming soon: SpaceEx the **state space** explorer (G. Frehse)
- A tool platform for developing hybrid verification tools
- Two tools will be released in 2010: PHAVer 2.0 for **linear hybrid automata** and a tool for **piecewise-linear differential equations** using **support function** representation
- Web interface





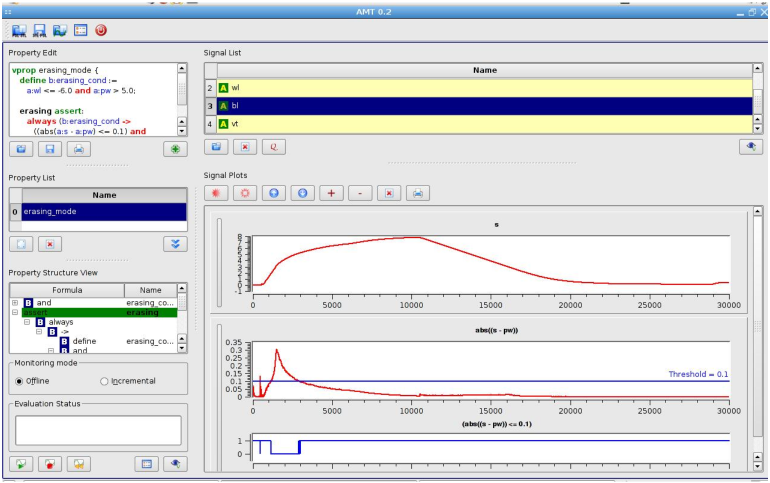
And What About Temporal Logic?

- The logic STL (**signal temporal logic**), an extension of the real-time logic MITL with numerical predicates
- Example: a **water-level controller** for a **nuclear plant** should maintain a controlled variable y around a fixed level despite external disturbances x
- We want y to stay always in the interval $[-30, 30]$ except, possibly, for an initialization period of duration 300
- If, due to disturbances, y goes outside the interval $[-0.5, 0.5]$, it should return to it within 150 time units and stay there for at least 20 time units
- The property is expressed as

$$\Box_{[300,2500]}((|y| \leq 30) \wedge ((|y| > 0.5) \Rightarrow \Diamond_{[0,150]} \Box_{[0,20]}(|y| \leq 0.5)))$$

Commercial II: AMT

- The **Analog Monitoring Tool** (D. Nickovic) is available for download



Thank You