Discrete and Hybrid Methods in Systems Biology

Oded Maler

CNRS - VERIMAG
Grenoble, France

SFBT 2012

Preamble

- Je ne suis pas un biologist et je vais parler en anglais so "theory" is my strongest link to this school

Preamble

- The intended messages in my talk are:
- 1) **Dynamical systems** are important for Biology
- 2) Those dynamical systems are **not** necessarily those that you learned about in school
- 3) Some inspiration for biological models should come more from **Informatics** and **Engineering** and less from **Physics**
- 4) In particular, methodologies for exploring the behavior of **under-determined** (open) dynamic models

Organization

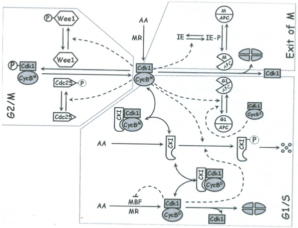
- Part I
  - Dynamical systems in Biology
  - Discrete-Event Dynamical Systems (Automata)
  - What is Verification
- Part II
  - Applying Verification to Continuous and Hybrid Systems
  - Parameter-Space Exploration
  - Reachability Computation

Dynamical Systems are Important

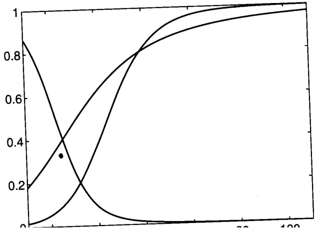
- Not news for biologists with a mathematical background
- J.J. Tyson, **Bringing cartoons to life**, Nature 445, 823, 2007:
-
- "Open any issue of *Nature* and you will find a diagram illustrating the molecular interactions purported to underlie some behavior of a living cell.
- The accompanying text explains how the link between molecules and behavior is thought to be made.
- For the simplest connections, such stories may be convincing, but as the *mechanisms* become more complex, *intuitive* explanations become more error prone and harder to believe."

In other Words

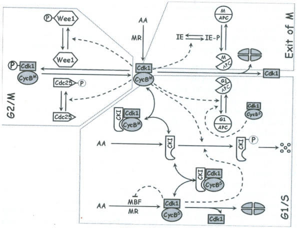
- What is the relation (if any) between

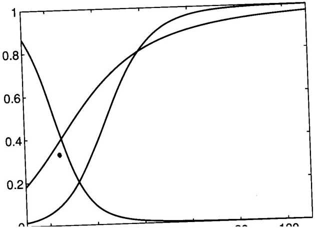


and



Systems and Behaviors





- Left: a *model* of a *dynamical system* which explains the mechanism in question
- Right: some *experimentally observed* behavior supposed to have some relation to the behaviors that the dynamical model generates
- What is this relation exactly?
- Current practice leaves a lot to be desired (at least for theoreticians)

An Illustrative Joke

- An *engineer*, a *physicist* and a *mathematician* are traveling in a train in Scottland. Suddenly they see a **black** sheep
- Hmmm, says the engineer, I didn't know that sheeps in Scottland are **black**
- No my friend, corrects him the physicist, *some* sheeps in Scottland are **black**
- To be more precise, says the mathematician, *there is* a sheep in Scottland having *at least one* **black** side

An Illustrative Joke

- A discipline is roughly characterized by the number of logical quantifiers $\exists$ $\forall$ (and their alternations) its members feel comfortable with

An Illustrative Joke

- By the way what would a biologist say?

An Illustrative Joke

- By the way what would a biologist say?
- In the Scottish sheep the agouti isoform is first expressed at E10.5 in neural crest-derived ventral cells of the second branchial arch

Dynamical Systems, a Good Idea

- The quote from Tyson goes on like this:
- “A better way to build bridges from **molecular biology** to **cell physiology** is to recognize that a network of interacting genes and proteins is ..
- .. a **dynamic** system evolving in space and time according to fundamental laws of reaction, diffusion and transport
- These **laws** govern how a regulatory network, confronted by any set of **stimuli**, determines the appropriate **response** of a cell
- This information processing system can be described in **precise** mathematical terms,

Dynamical Systems, a Good Idea

- These **laws** govern how a regulatory network, confronted by any set of **stimuli**, determines the appropriate **response** of a cell
- This information processing system can be described in **precise** mathematical terms,
- .. and the resulting equations can be **analyzed** and **simulated** to provide **reliable**, **testable** accounts of the molecular control of cell behavior"
- No news for engineers..

Models in Engineering

- To build complex systems other than by trial and error you need **models**
- Regardless of the language or tool used to build a model, at the end there is some kind of **dynamical system**
- A mathematical entity that generates **behaviors** which are progression of states and events in time
- Sometimes you can reason about such systems analytically

Models in Engineering

- Sometimes you can reason about such systems analytically
- But typically you **simulate** the model on the computer and generate behaviors
- If the model is related to **reality** you will learn **something** from the simulation about the **actual** behavior of the system

Models in Engineering

- Major difference: in engineering, the components are often well-understood and we need the simulation only because the outcome of their **interaction** is hard to predict

My Point: Systems Biology ≈ Dynamical Systems, but..

- To make progress in Systems Biology one needs to upgrade descriptive “models” by **dynamic models** with stronger predictive power and refutability
- Classical models of dynamical systems and classical analysis techniques tailored for them are **not** sufficient for effective modeling and analysis of biological phenomena

My Point: Systems Biology ≈ Dynamical Systems, but..

- Models, insights and computer-based analysis **tools** developed within **Informatics** (aka **Computer Science**) can help
- The whole systems thinking in CS is much more evolved and sophisticated than in physics and large parts of math
- This is true of other engineering disciplines such as circuit design or control systems

What “Is” Informatics ?

- Informatics is the study of **discrete-event dynamical systems** (automata, transition systems
- A natural point of view for for people working on modeling and verification of “**reactive systems**”
- Less so for data-intensive software developers and users

What "Is" Informatics ?

- This fact is sometimes **obscured** by fancy formalisms:
- Petri nets, process algebras, rewriting systems, temporal logics, Turing machines, programs
- All honorable topics with intrinsic beauty, sometimes even applications and deep insights

What “Is” Informatics ?

- All honorable topics with intrinsic beauty, sometimes even applications and deep insights
- But in an inter-disciplinary context they should be distilled to their **essence** to make sense to potential users..
- ..rather than **intimidate** them

Dynamical Systems in General

- The following abstract features of dynamical systems are common to both **continuous** and **discrete** systems:
- **State variables** whose set of **valuations** determine the **state space**
- A **time domain** along which these values evolve
- A **dynamic law**: **how** state variables evolve over time, possibly under the influence of **external** factors

Dynamical Systems in General

- A **dynamic law**: **how** state variables evolve over time, possibly under the influence of **external** factors
- System **behaviors** are **progressions** of **states** in **time**
- Knowing an initial state $x[0]$ the model can **predict**, to some extent, the value of $x[t]$

Types of Dynamical Systems

- Dynamic system models differ from each other according to their concrete details:
- State variables: numbers or more abstract types
- Time domain: metric (dense or discrete) or logical
- The form of the dynamical law (constrained, of course, by the state variables and time domain)
- The type of available analysis (analytic, simulation)
- Other features (open/closed, type of non-determinism, spatial extension)

Classical Dynamical Systems

- State variables: **real numbers** (location, velocity, energy, voltage, concentration)
- Time domain: the **real time axis** $\mathbb{R}$ or a discretization of it
- Dynamic law: **differential equations**

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

or their **discrete-time** approximations

$$x[t+1] = f(x[t], u[t])$$

Classical Dynamical Systems

- Dynamic law: **differential equations**

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

or their **discrete-time** approximations

$$x[t+1] = f(x[t], u[t])$$

- Behaviors: **trajectories** in the continuous state space
- Typically presented in the form of a collection of **waveforms**, mappings from time to the state-space
- What you would construct using tools like Matlab Simulink, Modelica, etc.

Discrete-Event Dynamical Systems (Automata)

- An **abstract discrete state space**
- State variables need **not** have a numerical meaning
- A **logical time domain** defined by the **events** (order but not metric)
- Dynamics defined by **transition rules**: input event **a** takes the system from state **s** to state **s**′

Discrete-Event Dynamical Systems (Automata)

- Dynamics defined by **transition rules**: input event **a** takes the system from state **s** to state **s**′
- Behaviors are **sequences** of **states** and/or **events**
- **Composition** of large systems from small ones using: different modes of **interaction**: synchronous/asynchronous, state-based/event-based
- What you will build using tools like Raphsody or Stateflow (or even C programs or digital HDL)

Preview: Timed and Hybrid Systems

- Mixing discrete and continuous dynamics
- **Hybrid automata**: automata with a different continuous dynamics in each state
- Transitions = mode switchings (valves, thermostats, gears, genes)

Preview: Timed and Hybrid Systems

- **Timed systems**: an intermediate level of abstraction
- Timed Behaviors = discrete events embedded in metric time, Boolean signals, Gantt charts
- Used implicitly by **everybody** doing real-time, scheduling, embedded, planning in professional **and** real life
- Formally: **timed automata** (automata with clock variables)

Automata: Modeling and Analysis

- Automata model processes viewed as **sequences** of **steps**: software, hardware, ATMs, user interfaces administrative procedures, cooking recipes, smart phones...
- Unlike continuous systems there are **no** simple analytical tools to determine long-term behavior
- We can **simulate** and sometimes do formal verification:

Automata: Modeling and Analysis

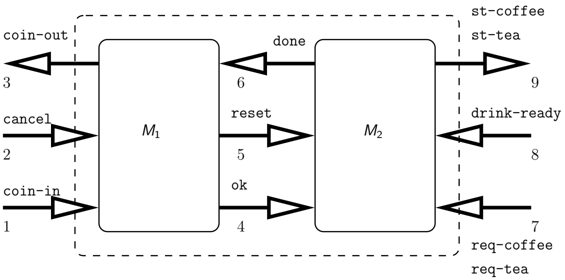
- We can **simulate** and sometimes do formal verification:
- Check whether **all** behaviors of a system, exposed to some uncontrolled inputs, exhibit some **qualitative** behavior:
- *Never reach some part of the state space; Always follow some sequential pattern of behavior...*

Automata: Modeling and Analysis

- *Never reach some part of the state space; Always follow some sequential pattern of behavior...*
- These **temporal properties** include **transients** and are much richer than classical **steady states** or **limit cycles**
- Tools for the verification of huge systems by sophisticated graph algorithms

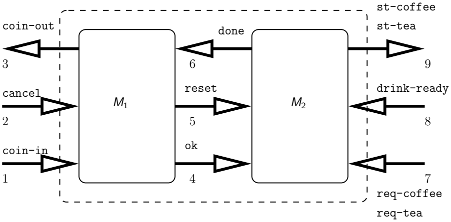
Illustration: The Coffee Machine

- Consider a machine that takes money and distributes drinks
- The system is built from two subsystems, one that takes care of financial matters, and one which handles choice and preparation of drinks
- They communicate by sending messages



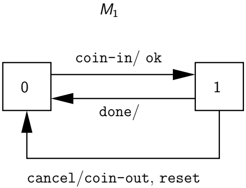
Remark: Signalling

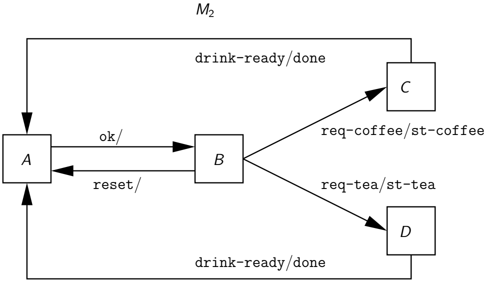
- Modern systems separate **information-processing** from the **physical interface**
- An inserted coin, a pushed button or a full cup are **physical events** translated by sensors into uniform low-energy signals
- These signals are treated as information, without thinking too much about their material realization
- Unless you are a low-level hardware designer



Automaton Models

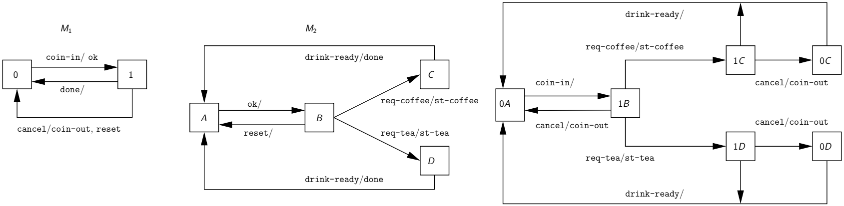
- The two systems are models as automata
- transitions are triggered by external events and events coming from the other subsystem



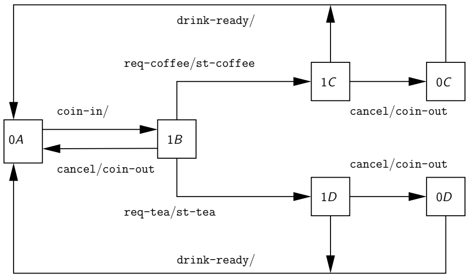


The Global Model

- The behavior of the whole system is captured by a composition (product) $M_1 \parallel M_2$ of the components
- States are elements of the Cartesian product of the respective sets of states, indicating the state of each component
- Some transitions are independent and some are synchronized, taken by the two components simultaneously
- Behaviors of the systems are paths in this transition graph



Normal Behaviors



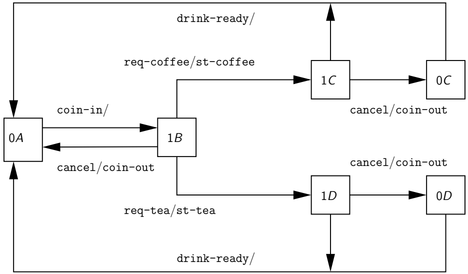
- Customer puts coin, then sees the bus arriving, cancels and gets the coin back

$$0A \; \texttt{coin-in} \; 1B \; \texttt{cancel} \; \texttt{coin-out} \; 0A$$

- Customer inserts coin, requests coffee, gets it and the systems returns to initial state

$$0A \; \texttt{coin-in} \; 1B \; \texttt{req-coffee} \; \texttt{st-coffee} \; 1C \; \texttt{drink-ready} \; 0A$$

An Abnormal Behavior



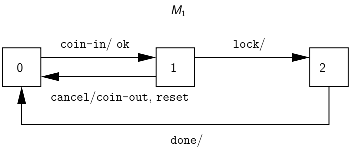
- Suppose the customer presses the cancel button *after* the coffee starts being prepared..

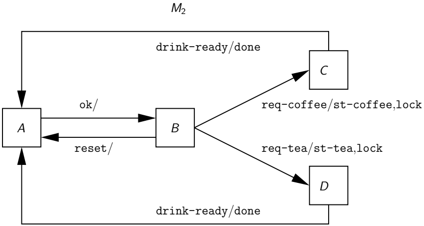
  $0A$ `coin-in` $1B$ `req-coffee` `st-coffee` $1C$ `cancel` `coin-out` $0C$ `drink-ready` $0A$

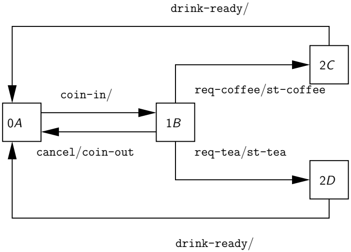
- Not so attractive for the owner of the machine

Fixing the Bug

- When M_2 starts preparing coffee it emits a `lock` signal
- When M_1 received this message it enters a new state where `cancel` is refused







The Moral of the Story I

- Many complex systems can be modeled as a composition of interacting automata
- Behaviors of the system correspond to paths in the global transition graph of the system
- The size of this graph is exponential in the number of components (state explosion, curse of dimensionality)

The Moral of the Story I

- These paths are labeled by **input** events representing influences of the **external** environment
- Each input sequence may generate a different behavior
- We want to make sure that a system responds correctly to **all** conceivable inputs
- That it behaves properly in any environment (robustness)

The Moral of the Story II

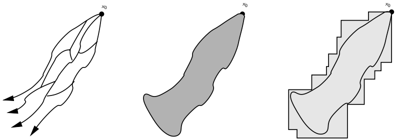
- How to ensure that a system behaves properly in the presence of all conceivable inputs and parameters?
- Each individual input **sequence** may induce a **different** behavior. We can **simulate** each but cannot do it exhaustively

The Moral of the Story II

- Verification is a collection of automatic and semi-automatic methods to analyze all the paths in the graph
- And this type of analysis and way of looking at phenomena is our **potential contribution** to Biology

Our Modest Contribution

- We develop analysis methods and **tools** that take under-determination seriously
- Either by **systematic sampling** of the uncertainty space
- Either by exhaustive **set-based** simulation methods that compute "tubes" of trajectories the contain **all** the behaviors under **all** choices in the uncertainty space



- and identifying the range of model parameters that lead to certain classes of behaviors
- Hopefully such tools will help increasing the meaningfulness of dynamic models and provide for their **composition**

Part II: Exploring Under-Determined Continuous Systems

- A system admits a dynamics $x[t+1] = f(x[t], p, u[t])$ where:
- p is a vector of **parameter** values
- Experiments do not characterize their exact values (they may vary among cells)
- $u[t]$ is an external disturbance signal indicating possible **dynamic** variations in the environment outside the model
- To generate a simulated behavior from an under-determined model you need to **fix**:
- **initial state** x_0, a **point** p in the parameter space, and a **disturbance profile** $u[t]$

Dynamical Models

- What does a simulator need to produce
- A **trace**:

$$x[0], x[1], x[2], \ldots$$

- For **deterministic** systems the dynamic rule is a function $f : X \to X$
- The rule allows the simulator to proceed from one state to another

$$x[i+1] = f(x[i])$$

- You just have to **fix** the initial state $x[0]$

Static/Punctual Under-Determination

- Some systems may have a **unique** initial state (reboot)
- Otherwise, to produce a trace you need to fix $x[0]$
- Without this information, the system is **under-determined** and **cannot** generate a trace
- It has an **empty slot** that needs to be filled by some **point** in $x \in X_0 \subseteq \mathbb{R}^n$, the set of all possible initial states
- Hence we call it **punctual** under-determination

Reminder: Models and Reality

- Whenever our models are supposed to represent something non-trivial they are just **approximations**
- This is evident for anybody working in modeling concrete **physical** systems
- It is less so for those working on the functionality of **digital hardware** or **software**
- There you have strong **deterministic** abstractions (logical gates, program instructions)
- A common way to pack our ignorance in a compact way is to introduce **parameters** ranging in some **parameter space**

Examples:

- **Biochemical reactions** in cells following the **mass action** law
- Many parameters related to the affinity between molecules
- Cannot be deduced from first principles, only measured by isolated experiments under different conditions

Examples:

- **Voltage level** modeling and simulation of circuits
- A lot of variability in transistor characteristics depending on production batch, place in the chip, temperature, etc.

Examples:

- **Timing performance analysis** of a new application (task graph) on a new multi-core architecture
- Precise execution times of tasks are not known before the application is written and the architecture is built

Parameterize Dynamical Systems

- The dynamics f becomes a **template** with some empty slots to be filled by parameter values
- Taken from some parameter space $P \subseteq \mathbb{R}^m$
- Each p instantiates f into a concrete function f_p that can be used to produce traces
- Parameters like initial states are instances of **punctual** under-determination: you choose them only once when starting the simulation

So What?

- So you have a model which is under-determined, or equivalently an **infinite** number of models
- For simulation you **need** to determine, to make a choice to pick a point p in the parameter space
- The simulation shows you something about **one** possible behavior of the system, or a behavior of **one** possible system
- But another choice of parameter values could have produced a completely different behavior
- Ho do you live with that?

Possible Attitudes

- The answer depends on many factors
- One is the **responsibility** of the modeler/simulator
- What are the consequences of not taking under-determination seriously
- Is there a penalty for jumping into conclusions based on one or few simulations?

Possible Attitudes

- Another factor is the mathematical and real natures of the system you are dealing with
- And as usual, it may depend on culture, background and tradition in the industrial or academic community

Non Responsibility: a Caricature

- Suppose you are a scientist not engineer, say biologist
- You conduct experiments and observe traces
- You propose a model and **tune** the parameters until you obtain a trace similar to the one observed experimentally
- These are **nominal** values of the parameters

Non Responsibility: a Caricature

- Then you can publish a paper about your model
- Except for picky reviewers there are no real consequences for neglecting under-determination
- The situation is different if some engineering is involved (pharmacokinetics, synthetic biology)
- Or if you want others to **compose** their models with yours

Justified Nominal Value

- You can get away with using a nominal value if your system is very **continuous** and **well-behaving**
- Points in the neighborhood of p generate **similar** traces
- There are also mathematical techniques (bifurcation diagrams, etc.) that can tell you sometimes what happens when you change parameters
- This smoothness is easily broken by mode switching

Justified Nominal Value

- Another justification for ignoring parameter variability:
- When the system is adaptive anyway to deviations from nominal behavior (control, feedback)

Taking Under-Determination More Seriously: Sampling

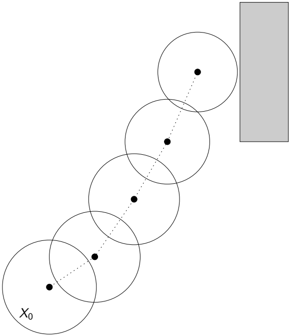
- One can **sample** the parameter space with or without probabilistic assumptions
- Make a grid in the parameter space (exponential in the number of parameters)
- Or pick parameter values at random according to some distribution

Taking Under-Determination More Seriously: Sampling

- In the sequel I illustrate a technique (due to **A. Donze**) for adaptive search in the parameter space
- Sensitivity information from the numerical simulator tells you where to **refine** the coverage
- Arbitrary dimensionality of the state space, but no miracles against the dimensionality of the parameter space

Sensitivity-based Exploration I

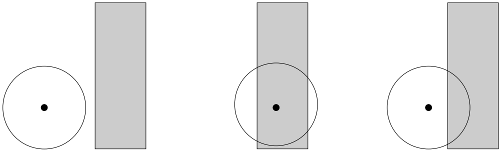
- We want to prove **all** trajectories from X_0 do not reach a bad set of states
- Take $x_0 \in X_0$ and build a ball B_0 around it that covers X_0



- Simulate from x_0 and generate a sequence of balls $B_0, B_1, \ldots$
- B_i **contains** all points reachable from B_0 in i steps

Sensitivity-based Exploration II

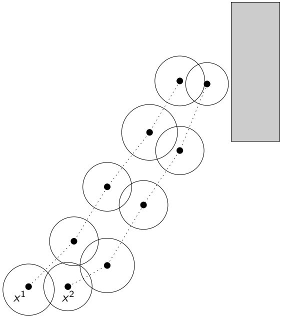
- After k steps, three things may happen:



- 1. No ball intersects bad set and the system is **safe** (over-approximation)
- 2. The concrete trajectory intersects the bad set and the system is **unsafe**
- 3. Ball B_k intersects the bad set but we do not know if it is a real or spurious behavior

Sensitivity-based Exploration III

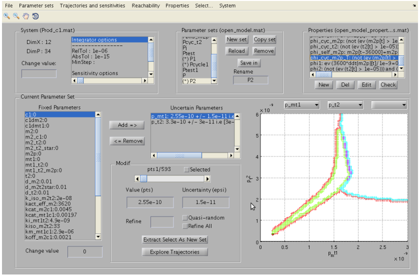
- In the latter case we refine the coverage and repeat the process for two **smaller** balls

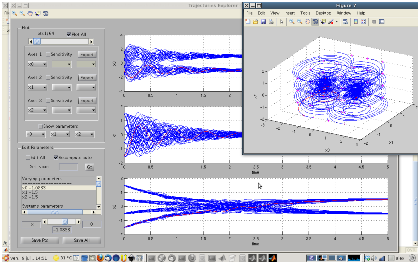


- Can prove correctness using a **finite** number of simulations, focusing on the interesting values
- Can approximate the boundary between parameter values that yield some qualitative behaviors and values that do not

The Breach Toolboox

- Parameter-space exploration for arbitrary continuous dynamical systems relative to **quantitative temporal properties**
- Applied to embedded control systems, analog circuits, biochemical reactions
- Available for download





Dynamic Under-Determination

- The system is modeled as **open**, exposed to external disturbances
- Dynamics of the form

$$x[i+1] = f(x[i], v[i])$$

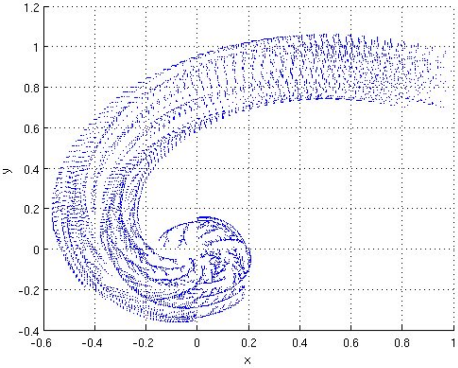
- The natural way to represent the influence of other unmodeled subsystems and the external environment

Dynamic Under-Determination

- Under-determination becomes dynamic: to produce a trace you need to give the value of v at every step in time, a signal/sequence $v[1], \ldots, v[k]$
- A priory a much larger space to sample from: dimension mk compared to m for static
- One can use a nominal value: constant, step, periodic signal, random noise, etc.

Taking Under-Determination More Seriously: Sampling

- A method due to **T. Dang**:
- Use ideas from robotic motion planning (RRT) to generate inputs that yield a good **coverage** of the reachable state space
- Applied to analog circuits

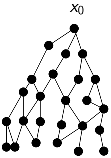


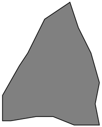
Taking Under-Determination More Seriously: Verification

- Paranoid **worst-case** formal verification attitude:
- If we say something about the system it should be provably true for **all** choices of p, $x[0]$ and $v[1], \ldots, v[k]$
- Instead of doing a simple simulation you do **set-based** simulation, computing **tubes of trajectories** covering everything

Taking Under-Determination More Seriously: Verification

- Instead of doing a simple simulation you do **set-based** simulation, computing **tubes of trajectories** covering everything
- Breadth-first rather than depth-first exploration



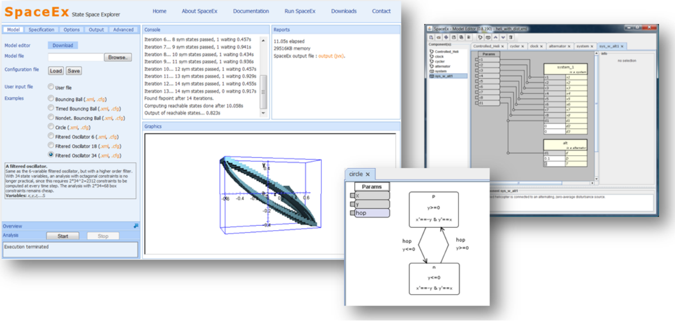


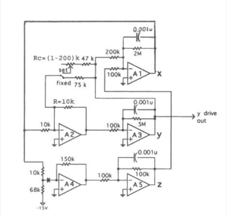
- Advantages: works also for hybrid (switched) systems
- Limitations: manipulates geometric objects in high dimension

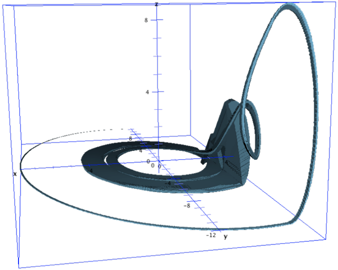
State of the Art

- Linear and piecewise-linear dynamics ~ 200 variables using algorithms of **C. Le Guernic and A. Girard**
- Nonlinear dynamics with $10 - 20$ variables - an ongoing research activity
- Implemented into the **SpaceEx** tool developed under the direction of **G. Frehse**
- Available on `http://spaceex.imag.fr` with web interface, model editor, visualization and more
- Waiting for more beta testers

The State-Space Explorer (SpaceEx)

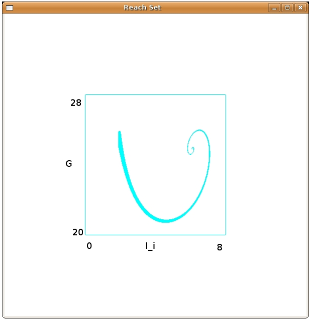


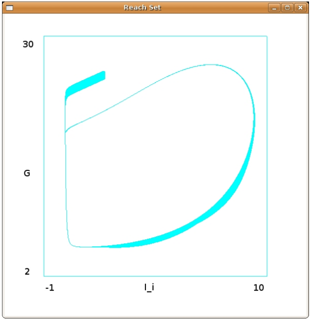




Example Lac Operon (T. Dang)

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





Back to the Big Picture

- Biology needs (among other things) more dynamic models to form verifiable predictions
- These models can benefit from the accumulated understanding of dynamical system within informatics and cannot rely only on 19th century mathematics
- The views of dynamical system developed within informatics are, sometimes, more adapted to the complexity and heterogeneity of Biological phenomena

Back to the Big Picture

- Biological modeling should be founded on various types of dynamical models: continuous, discrete, hybrid and timed
- These models should be strongly supported by computerized analysis tools offering a range of capabilities from simulation to verification and synthesis

Back to the Big Picture

- Systems Biology should combine insights from:
- Engineering disciplines: modeling and analysis of very complex man-made systems (chips, control systems, software, networks, cars, airplanes, chemical plants)
- Physics: experience in mathematical modeling of natural systems with measurement constraints
- Mathematics and Informatics as a unifying theoretical framework

Thank You