

Introduction

Computational Models for Complex Systems

Paolo Milazzo

Dipartimento di Informatica, Università di Pisa

<http://pages.di.unipi.it/milazzo>

milazzo@di.unipi.it

Laurea Magistrale in Informatica

A.Y. 2018/2019

About this course

General info:

- Course web page
<http://pages.di.unipi.it/milazzo/teaching/AA1819-CMCS/>
- Teaching hours (Aula Fib L1)
 - ▶ Wed 9-11
 - ▶ Fri 9-11
- Office hours
 - ▶ should be Tue 14-16, but
 - ▶ better if we fix an **appointment** via email...

About this course

Teaching info:

- Contents of this course
 - ▶ Methods for the modeling, simulation and formal analysis of complex systems such as populations, biological systems, social networks, markets, etc...
- Textbooks and software
 - ▶ No unique textbook and software tool.
 - ▶ A number of **lecture notes** and references to **useful software tools** will be made available on the course web page.
 - ▶ Also the **slides** of the course will be made available on the web page
- Exam
 - ▶ It may consist in either a **presentation** of a scientific paper, a **small project** (to be done in groups of 1-3 students), or a standard **oral exam**.
 - ▶ Proposals of scientific papers and projects ideas will be made available close to the end of the course (proposals from students are welcome!)
 - ▶ If possible, a first round of student presentations will be organized **before the end** of the course (last couple of lessons)

Computational Models for Complex Systems

Computational models? Complex systems?

WHAT ARE WE TALKING ABOUT??

Let's start from understanding what, in general, is a

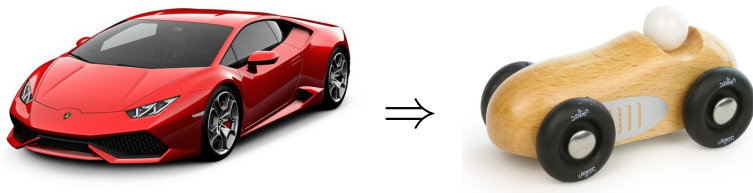
MODEL of a SYSTEM

Understanding requires modeling

Assume we want to

- **understand** how a given system works
- or that we want to **design** a new system.

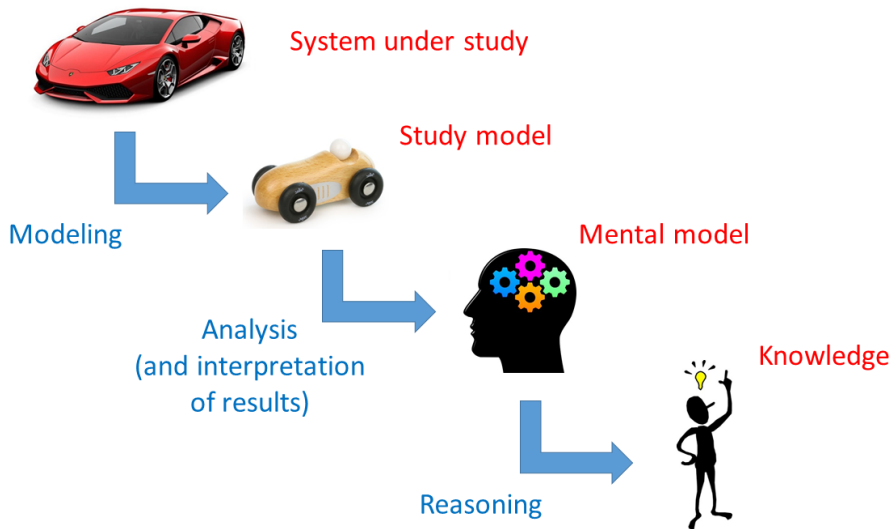
First of all, we need to build a model!



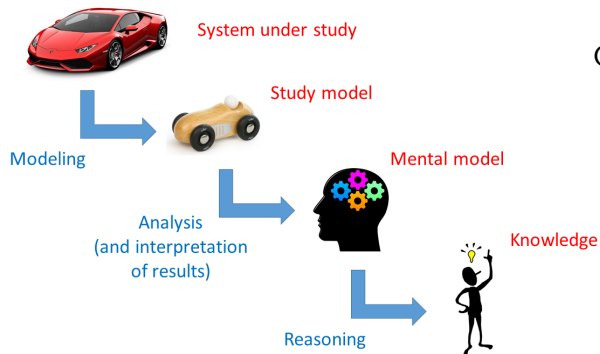
Definition: Model

A model is a **simplified and approximate representation** of a system, that allows reasoning on the system's properties

Understanding requires modeling



Sources of error



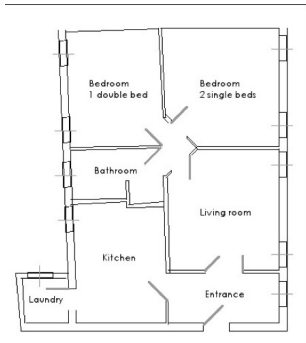
Common sources of error:

- Model inaccuracy
- Analysis method inaccuracy
- Analysis results ambiguity
- Inaccurate reasoning

Inaccuracies and ambiguities can be limited by adopting well-established scientific methods and rigorous (logical) reasoning techniques

Examples of models

From Architecture:



Planimetries/projects

Scale models

These are models of **artificial entities** (buildings) useful to assess structural properties at **design time**, and to let the buyer evaluate the result in advance

Examples of models

From Architecture:



Renderings



Virtual Reality

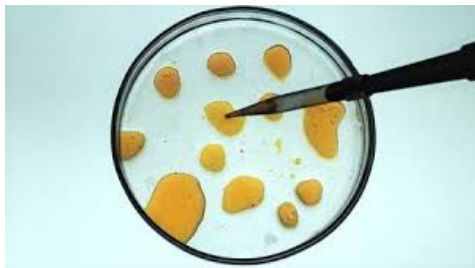
Computer-based techniques are nowadays **more accurate and realistic** than traditional approaches, and offer advanced interaction possibilities

Examples of models

From Life Sciences:



In vivo models
(mice, rats, pigs, ...)



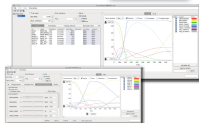
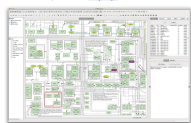
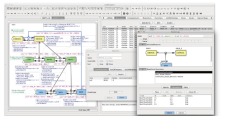
In vitro models
(cells in a test tube or Petri dish)

These are **models of the human being**, typically used to create **new knowledge** (on diseases, toxicity effects, etc...)

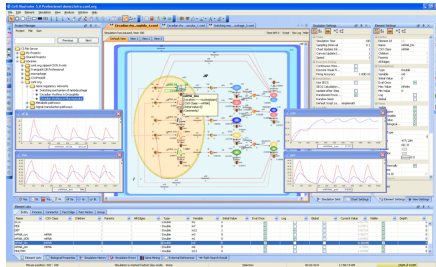
Examples of models

From Life Sciences:

Pathway Modeling



Simulation



In silico models

In silico models

Computer-based techniques are usually **faster and cheaper** than *in vivo* and *in vitro* methods, and sometimes they have comparable accuracy.

Mathematical modeling

Mathematics provides tools for building abstract models of almost everything

- geometry for Architecture
- differential equations for Weather

Properties of mathematical models (corresponding to properties of the modeled system) can be studied either

- analytically
(e.g. the volume of a building, or the payload of a bridge, can be computed precisely)
- or numerically
(e.g. weather forecasts, are based on approximate simulations)

Numerical analysis of mathematical models is quite similar to computational modeling

Computational modeling

Definition: Computational Model

A computational model is a **mathematical representation** of a **dynamical system** taking a **computer-executable** form.

First of all, with this definition we are **restricting** the class of systems of interest to **dynamical systems** (systems which evolve over time).

- So, e.g. the computer-based architectural models are not computational models, since building are not dynamical systems.

Then, a mathematical representation is necessary, since mathematics provides **unambiguous** (formal) tools to represent systems.

- An ambiguous model could be interpreted differently by the modeler and the computer

Computer-executable means that the computer should be able to **compute the evolution** of the system over time (e.g. by performing **simulations**)

Focus on dynamical systems

Dynamical systems are systems that **evolve** by changing their state over time

The state of a dynamical system is usually represented by a finite set of variables called **state variables**

- temperature, humidity, price, position, . . .

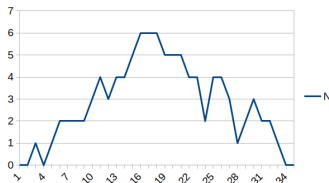
The state of the system (i.e. the values of the variables) can change in either a **discrete** or a **continuous** way (or both)

Focus on dynamical systems

Some simple examples:

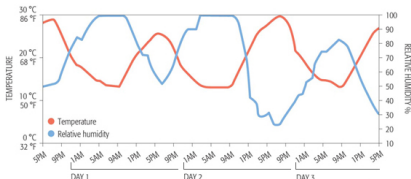
- Queue at the post office.

State variable: N (number of enqueued people)



- Weather.

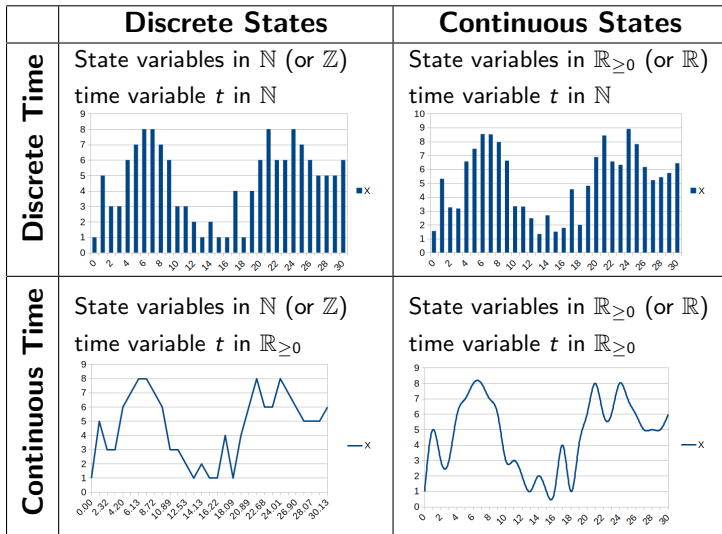
State variables: T (temperature), H (humidity)



© Pearson Education, Inc.

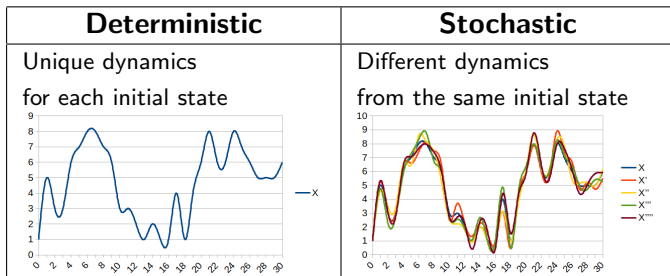
Focus on dynamical systems

Dynamical system span over a discrete/continuous spectrum



Focus on dynamical systems

Moreover, dynamical systems (and/or their models) can be deterministic or probabilistic/stochastic



Different types of systems should be modeled with different types of models

- Otherwise, **approximations** come into play

Why should we model dynamical systems?

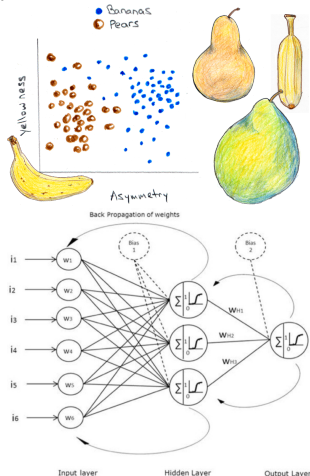
Models of dynamical systems can be used to study (on a computer)

- **Reachability of states** (predictions)
 - ▶ Will it rain tomorrow?
 - ▶ Will this drug cause toxicity?
 - ▶ Will vaccinations eradicate the disease?
 - ▶ Will this content spread in a social network?
- **Behavioral patterns**
 - ▶ Will the concentration of this substance become stable?
 - ▶ Will the size of the population oscillate?
 - ▶ Will cars stay out of the city center?
- **Effects of perturbations/control strategies**
 - ▶ What happens to the disease if we change the dosage of a drug?
 - ▶ What happens to fish populations if we stop fishing for one month?
 - ▶ Which impact would have the closure of a street?

How to build a computational model?

The **Data-driven** way (e.g. Machine Learning)

- The model is directly **inferred from data**
- Usually requires **a lot of data**, but **limited knowledge** about the system functioning
- The model takes a form suitable for inference method used (e.g. a Neural Network)
- If enough data is available, it often works very well (**good predictions**)
- Inferred models are often **very difficult to be interpreted**



How to build a computational model?

The **Knowledge-driven** way (e.g. **this course!**)

- The model is constructed **on the basis of the available knowledge** about the system's internal mechanisms
- Usually requires **limited data**, but **a good knowledge** about the system functioning
- The model takes a form suitable to describe the system's internal mechanisms
- Model construction usually some effort, and often predictions suffer from approximations, but
 - ▶ the method works also when few data are available
 - ▶ the model is interpretable: it contributes to **understanding why** a system behaves as observed
 - ▶ modeling allows **validating hypotheses** on the system functioning

Complex systems

Putting the pieces together, we have more or less understood what is a

COMPUTATIONAL MODEL of a **DYNAMICAL SYSTEM**

but this course deals with **COMPLEX** systems... what do they are?

Complex systems a particular class of dynamical systems...

Complex systems

Definition: Complex system

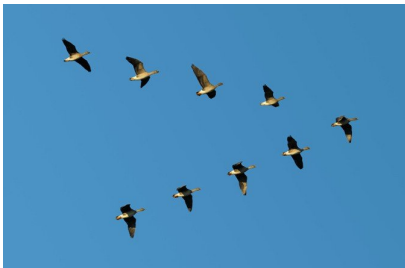
A complex system is a system consisting of **many simple components interacting with each other**, from whom a **collective behavior** with interesting dynamical properties **emerges**.

Complex system's behavior out of **simple component interactions**

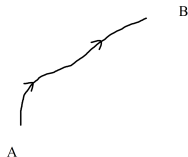
- Holistic interpretation: **“the whole is more than the sum of the parts”**
- The behavior of the system **emerges**, it is not easily predictable from the knowledge of the individual components

Examples of complex systems

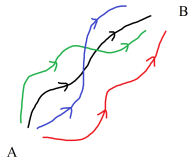
Flocking of birds



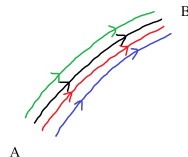
One bird



Another bird



Four separate birds



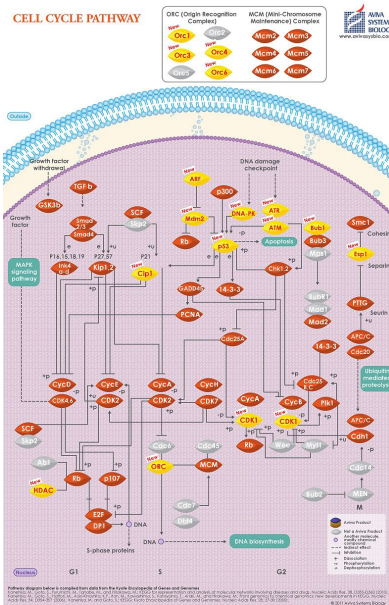
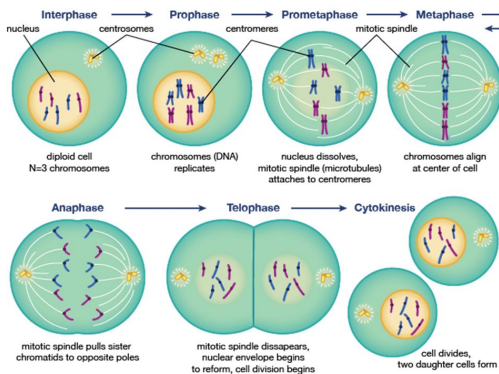
Four birds in a flock

Examples of complex systems

In the flocking example:

- the **complex system** is the **flock**
- the simple **components** are the **birds**
- the **individual behavior** of each component is
“fly towards destination”
- the **interactive behavior** of each component is
“fly close to the other birds, follow the leader”
- the **emergent behavior** (of the flock) is
“maintain the typical V-shape”

The cell cycle
(process leading to cell duplication)



Examples of complex systems

In the cell cycle example:

- the **complex system** is the **cell**
- the simple **components** are the **biomolecules** (DNA, proteins, etc...)
- the **individual behavior** of each component is
“be synthesized and degraded”
- the **interactive behavior** of each component is
“participate in chemical reactions”
- the **emergent behavior** (of the cell) is
“growth and division into two cells”

Examples of complex systems

Ants foraging (search for food)



When a colony of ants finds a source of food, they organize in a trail along the path from their nest to the food source

This is obtained through a mechanism called **stigmergy**

- **indirect coordination** mechanism: ants leave drops of a chemical along the path, other ants follow such a trace and reinforce it by leaving additional drops

Examples of complex systems

In the ants foraging example:

- the **complex system** is the **ant colony**
- the simple **components** are the **ants**
- the **individual behavior** of each component is
“go around, looking for food”
- the **interactive behavior** of each component is
“leave, detect and follow drops of chemicals”
- the **emergent behavior** (of the colony) is
“trail formation”

Differently from the birds and biomolecules in the previous examples, ants communication is not direct, but **mediated by the environment** (where drops are left)

Examples of complex systems

Traffic



Examples of complex systems

In the traffic example:

- the **complex system** is the **urban mobility system**
- the simple **components** are the **cars**
- the **individual behavior** of each component is
“drive to destination, according to the rules”
- the **interactive behavior** of each component is
“avoid crashes”
- the **emergent behavior** (of the colony) is
“queues”

Again, the **environment** plays a role. Streets are a **limited resource** that is shared by cars

- when the number of cars exceeds the street capacity, queues start to form

Examples of complex systems

Social networks



Examples of complex systems

In the social network example:

- the **complex system** is the **network**
- the simple **components** are the **members (profiles)**
- the **individual behavior** of each component is
“expose your profile”
- the **interactive behavior** of each component is
“share and look at other's shares, communicate”
- the **emergent behavior** (of the network) is
“trend topics, formation of communities, ...”

These networks are **dynamic**

- their structure (connections) evolve over time
- **network science** is the discipline that studies the evolution of network structures
- in **complex networks**, interesting global topologies emerge from local changes (like in complex systems)

Modeling notations for complex systems

Many **modeling languages** are available for complex systems

From mathematics:

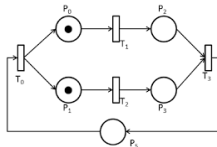
- Recurrence relations
- Differential equations

$$X_t = 2X_{t-1}Y_{t-1} - 3$$

$$\dot{X} = 2XY - 3$$

From concurrency theory:

- Petri nets
- Multiset rewriting
- Process calculi

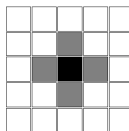


$$A + B \rightarrow C$$

$$a.P \mid \bar{a}.Q$$

From artificial life:

- Cellular automata
- Agents



```
to eat-grass
  ask turtles [
    if pcolor = green [
      set pcolor black
      set energy energy + 10
    ]
  ]
end
```

From modeling notations to behaviors

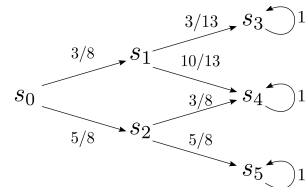
Modeling languages allow the modeler to express

- the **relationships** between the state variables of the system,
- and the **rules/laws** that determine the change of their values over time

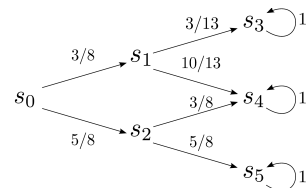
The **dynamics** (or behavior) of the system (the actual sequences of states reached by the system over time) can be computed according to the **semantics** of the modeling language

- Semantics gives a meaning to a language

The semantics of a modeling language can often be expressed as a **transition system**



Analysis of behaviors



Once you have a description of the system's behavior (the transition system) you can **analyze** it in several ways:

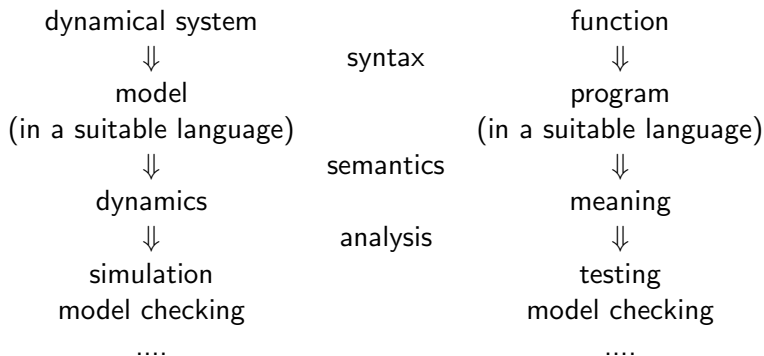
- **simulation** (computes a trace of the transition system)
- **model checking** (determine whether the whole transition system satisfies a given dynamical property)
- ... static analysis, steady state analysis, sensitivity analysis, ...

Simulation does not require constructing the whole transition system

Modeling vs programming

Modeling and programming are very similar

- a program is somehow a model of a function



Workplan

This is the (tentative) schedule of this course:

- Mathematical modeling of dynamical systems
 - ▶ Discrete models (recurrence equations)
 - ▶ Continuous models (ordinary differential equations - ODEs)
- The chemical reaction metaphor
 - ▶ From ODEs to stochastic simulation
- Transition systems and model checking
- Modeling notations:
 - ▶ Multiset rewriting
 - ▶ Petri nets
- Discrete event simulation
- Structured approaches:
 - ▶ Agent-based modeling and simulation
 - ▶ Interactions over a network

Workplan

We will see applications to:

- social behaviors of animals (e.g. prey-predator interaction, flocking)
- epidemiology (e.g. spread of diseases and vaccination)
- biology (e.g. cell pathways)
- social systems
- ...

Software

Software we will use:

- your favorite spreadsheet (Excel, Openoffice calc, Google docs, ...)
- GNU Octave (<https://www.gnu.org/software/octave/>)
- Dizzy stochastic simulator (available on the course web page)
- PRISM Model Checker (<https://www.prismmodelchecker.org/>)
- NetLogo (<https://ccl.northwestern.edu/netlogo/>)
- Java (or any other programming language...)