On Simulating Complex Computing Systems

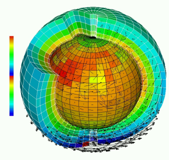
Frédéric Suter

March 11, 2015
FJPPL Computing Workshop

What is Science?

Doing Science = Acquiring Knowledge

$$\frac{\partial}{\partial x_j}\left(\frac{\partial \Phi}{\partial x_i}\right) = \frac{\partial}{\partial x_i}\left(\frac{\partial \Phi}{\partial x_j}\right)$$



Experimental Science

- Thousand years ago
- Observations-based
- Can describe ...
- ... but not predict

Theoretical Science

- Last few centuries
- Equations-based
- Can understand
- Prediction long

Computational Science

- Nowadays
- Compute-intensive
- Can simulate
- Prediction easier

Science is Still Experimental

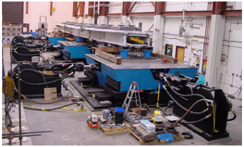
Space telescope

Large Hadron Collider

Mars Explorer

Tsunamis

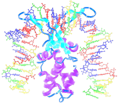
Earthquake vs. Bridge

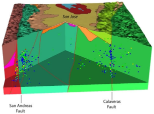


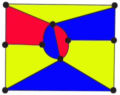
Climate vs. Ecosystems

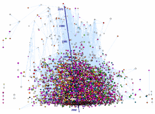
(who said that science is not fun??)

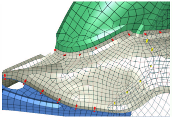
Computational Science

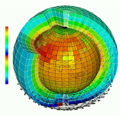




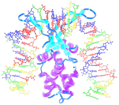


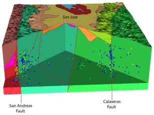


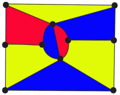


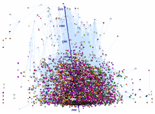


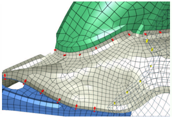
Computational Science

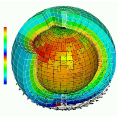




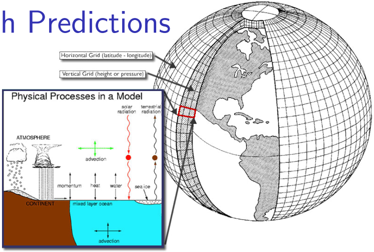




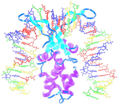


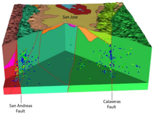


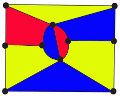
Understanding the Climate Change with Predictions

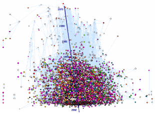


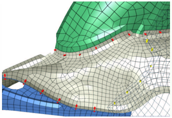
Computational Science

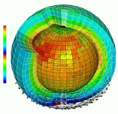




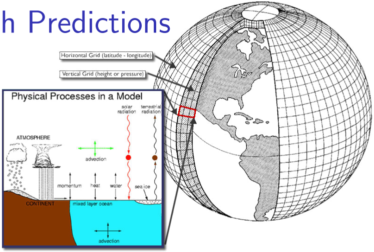




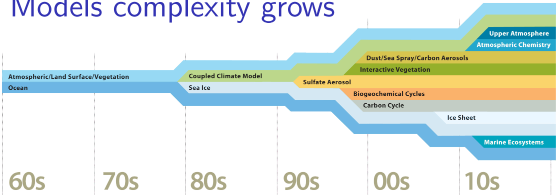




Understanding the Climate Change with Predictions



Models complexity grows



This requires
LARGE computers

How Large?

Massive parallelism

- Impossible to further miniaturize (atomic limit)
- Impossible to increase frequeny (energy)
- Solution: Multiply the number of cores!
- Tianhe-2, Top500 #1: 3,120,000 cores

Toward Exascale

- 1 Exaflop per second systems in 2020
- 1 Exaflop = 10^{18} operations. One million of millions of millions of operations...
  A human speed, 10 times the age of the Universe

Not limited to HPC

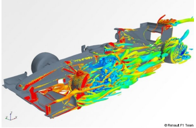
- Google computers dissipate 300MW on average (150,000 households, $\frac{1}{3}$ reactor)
- Botnets: BredoLab estimated to control 30 millions of zombie computers
- All these systems are heterogeneous and dynamic

How can we *study* such computing systems?

Assessing Distributed Applications/Systems

Performance Study $\rightsquigarrow$ Experimentation

- Maths: Often not sufficient to fully understand these systems



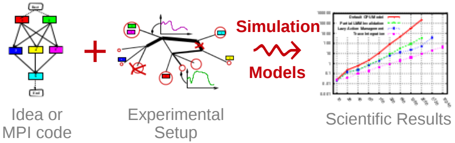
Courtesy of Lucas Nussbaum

- Experimental Facilities: <u>Real</u> applications on <u>Real</u> platform *(in vivo)*
- Emulation: <u>Real</u> applications on <u>Synthetic</u> platforms *(in vitro)*
- Simulation: <u>Prototypes</u> of applications on system's <u>Models</u> *(in silico)*

Simulating Complex Distributed Systems

Fastest path from idea to data

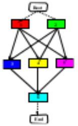
- Get preliminary results from partial implementations
- Experimental campaign with thousands of runs within a week
- Test your scientific idea, don't fiddle with technical subtleties (yet)



Easiest way to study distributed applications

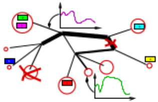
- Everything is actually centralized: Partially mock parts of your protocol
- No heisenbug: (Simulated) time does not change when you capture more data
- Clairevoyance: Observe every bits of your application and platform
- High Reproducibility: No or very few variability
- *What-if?* exploration: Can we remove/add/upgrade a component?
- Ecological: Don't waste resources for debug and test

Simulation Challenges



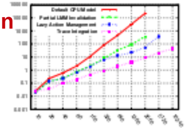
Idea or
MPI code

+

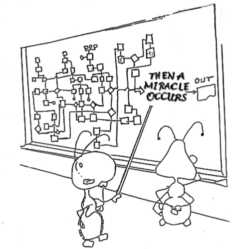


Experimental
Setup

Simulation
⇝
Models



Scientific Results



Challenges for the Tool Makers

- Validity: Get realistic results (controlled experimental bias)
- Scalability: *Fast enough* and *Big enough*
- Tooling: runner, post-processing, integrated lab notes
- Applicability: Simulate what is important to the user

Major Components of any Simulation-based Experiment

- An observation of the application: either a trace or the live application
- Models of the platform: CPU, network, any other relevant resource
- A configuration describing the experimental settings

SimGrid: a Versatile Simulation Toolkit

Scientific Instrument

- Versatile: Grid, P2P, HPC, Volunteer Computing and others
- Sound: Validated, Scalable, Usable; Modular; Portable
- Community-driven: 30 contributors (5 not affiliated), 5 contributed tools, GPL

Scientific Object

- Comparison of network models on non-trivial applications
- High-Performance Simulation on realistic workload
- Full model checker (ongoing emulation support)

Sustained Project

- Impact: 120 publications (110 distinct authors, 5 continents), 4 Ph.D.
- Started in 1998 at UCSD, now an international collaboration
- 7 partners, 20+ researchers (CNRS, Universities, Inria)
- Public funding (≈3M€ ANR/Inria)

Simulation in a HEP Computing Center?

Several Motivating Use Cases

- Dimensioning the Computing Infrastructure
  - Prepare future upgrades
- Understanding the Hierarchical Mass Storage System
  - Prepare pledge discussions
- Having a Realistic Description of Production Grids
  - Optimize incoming workload
- . . .

Disclaimer

- CC-IN2P3's team: 2-3 people only
  - $\Rightarrow$ Hardly possible to address all the use cases
- SimGrid is not a simulator but a toolkit
  - We provide the abstractions, models, and APIs
  - It's up to users to embrace the tool and conduct studies

Dimensioning the Computing Infrastructure

The Problem

- Data centers have to increase their capacities
  - More CPUs, more storage, and faster networks
- A bad desicion may have a high cost
  - Cannot wait for implementation to know the good/bad outcome

A True Dimensioning Story

- How should the compute cluster be upgraded?
  - More cores or a high performance network?
- Rely on expertise of system administrators and/or users
  - Users were asked for their preference

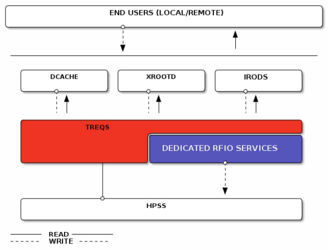
Simulation at Work

- Empirical decisions ⇒ Exploration of “what-if” scenarios
- Subjective perception ⇒ Objective indicators

Understanding the Hierarchical Mass Storage System

The Problem

- Complex stack of software and protocols
  - HPSS / RFIO
    - Clients ↔ Disks ↔ Tapes
  - TReqs ↔ HPSS
  - DCache/XrootD/IRods ↔ TReqs↔ HPSS
  - Users ↔ DCache/XrootD ↔ TReqs ↔ HPSS
- Difficult to measure the impact of a decision at a given level



Simulation at Work

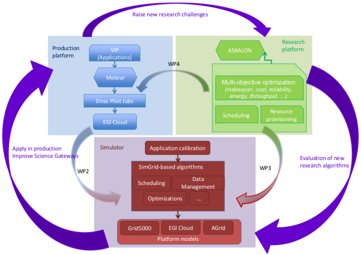
- Definition, design and implementation of a SimGrid's storage API
- Develop models and simulators at each level
  - Following a bottom-up approach

Realistic Description of Production Grids

The Problem

- VIP Scientific Gateway: Ease data sharing and access to computing resources
  - For medical image simulation: MRI, PET scans, CT scans
- Efficient scheduling of Monte-Carlo simulation on Grids
  - Production platform ⇝ High Variability ⇝ Performance assessment issues
  - Lack of applicative view of the platform

Simulation at Work



- Objectives
  - Reduce time to result
  - Scheduling heuristics
  - Simulated applicative view
- Method
  - Extract information from traces
    - Characterize and model
  - Inject dynamic conditions
  - Handle storage components

Take Away Messages

SimGrid could prove helpful to computing centers

- Versatile: Used in several communities (scheduling, Grids, HPC, P2P, Clouds)
- Accurate: Model limits known thanks to validation studies
- Sound: Easy to use, extensible, fast to execute, scalable to death, well tested
- Open: LGPL; User-community much larger than contributors group
- There for 15 years, and ready for at least 15 more years

Welcome to the Age of (Sound) Computational Science

SIMGRID

- Discover: `http://simgrid.org/`
- Learn: 101 tutorials, user manual, and examples
- Join: user mailing list, #simgrid on irc.debian.org