Expérimentations et calculs Distribués à Grande Échelle

Projet EDGE du CPER MISN

Lead by Martin Quinson, Lucas Nussbaum

LORIA

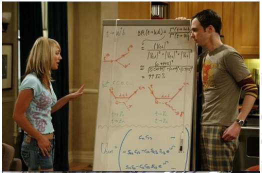
10 juin 2011

How does Science work?

Proposed theories remain valid until proved false (or better proposed)

Classical approaches in science and engineering

1. Theoretical work: equations on a board
2. Experimental study on an scientific instrument



The Big Bang Theory

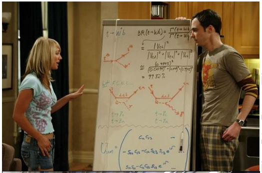
Large Hardron Collider

How does Science work?

Proposed theories remain valid until proved false (or better proposed)

Classical approaches in science and engineering

1. Theoretical work: equations on a board
2. Experimental study on an scientific instrument

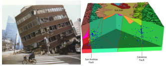


The Big Bang Theory

Large Hardron Collider

Not always desirable / possible

- Some phenomenons are intractable theoretically
- Experiments too expensive, difficult, slow, dangerous

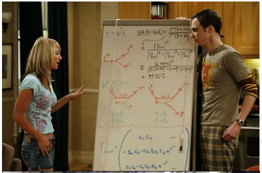


How does Science work?

Proposed theories remain valid until proved false (or better proposed)

Classical approaches in science and engineering

1. Theoretical work: equations on a board
2. Experimental study on an scientific instrument

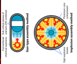


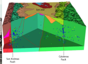
The Big Bang Theory

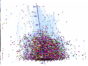
Large Hadron Collider

Not always desirable / possible

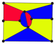
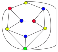
- Some phenomenons are intractable theoretically
- Experiments too expensive, difficult, slow, dangerous

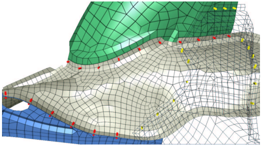








The third scientific way: *Computational Science*

3. Use computers (*in silico* study)
    - Modeling / Simulation of the phenomenon
    - Data Mining to find interesting subject of studies
    - Automated theorem proving  



Car Mesh

Our Scientific Objects: Distributed Systems

Scientific Computing: High Performance Computing / Computational Grids

- Infrastructure underlying *Computational science*: Massive / Federated systems
- Main issues: Have the world's biggest one / compatibility, trust, accountability

Cloud Computing

- Large infrastructures underlying commercial Internet (eBay, Amazon, Google)
- Main issues: Optimize costs; Keep up with the load (flash crowds)

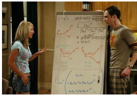
P2P Systems

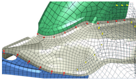
- Exploit resources at network edges (storage, CPU, human presence)
- Main issues: Intermittent connectivity (churn); Network locality; Anonymity

Systems already in use, but characteristics hard to assess

- Performance: everyone want to maximize it, but definition differs
- Correction: absence of crash, race conditions, deadlocks and other defects

Assessing Distributed Applications





Classical Scientific Pillars Apply

- Theoretical Approach: Mathematical study of algorithms
- Experimental Science: Study applications on scientific instrument
- Computational Science: Simulation of a system model

Performance Study ⇝ Experimentation

- Maths: these artificial artifacts contain what we've put in it
  But complex, dynamic, heterogeneous, scale ⇝ beyond our capacities
- Experimental Facilities: Real applications on Real platform *(in vivo)*
- Emulation: Real applications on Synthetic platforms *(in vitro)*
- Simulation: Prototypes of applications on system's Models *(in silico)*

	Experimental Facilities	Emulation	Simulation
Experimental Bias	☺☺	☺	☹
Experimental Control	☹☹	☺	☺☺
Ease of Use	☹	☹☹	☺☺

- Correction Study ⇝ Formal Methods (model-checking, proof, static evaluation)

EDGE Project

Experimentation and Distributed systems at Large Scale

1. Experimental methodologies for large scale computer systems
   - Facilities: large scale administration
   - Emulation: WrekAvoc and Simterpose projects
   - Simulation: SimGrid project
   - Overall Organization: Scalable Laboratory
2. Reasoned usage of modern computational platforms
   - Applications:
     - Analysis of crypto-systems
     - Theorem provers
     - P2P systems
   - Animation of the community:
     - Project Bootstraping (first year in CPER)
     - Towards academics
     - Towards industries
     - Towards production grids

EDGE Project

Experimentation and Distributed systems at Large Scale

1. Experimental methodologies for large scale computer systems
   - Facilities: large scale administration (new: graphene cluster – many nodes)
   - Emulation: WrekAvoc (new: rewrite) and Simterpose (new: feasibility)
   - Simulation: SimGrid project (new: SMPI now mature)
   - Overall Organization: Scalable Laboratory (new operation)
2. Reasoned usage of modern computational platforms
   - Applications:
     - Analysis of crypto-systems (new: RSA 768 broken)
     - Theorem provers (new: testing framework $\rightsquigarrow$ prover comparison/competitions)
     - P2P systems (new: evaluation of a P2P wiki with concurrent editing)
   - Animation of the community:
     - Project Bootstraping (first year in CPER)
     - Towards academics (new: user day / courses targeting prospective users)
     - Towards industries (new: ad hoc intl; Accelor-Mital – in vain so far)
     - Towards production grids (new: study EGEE on Grid'5000; contact INRA)

NOW: 2 focuses on 2010 work and one future project

Emulation as an Experimental Methodology

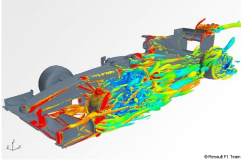
Execute real application in a perfectly controlled environment

- Real platforms are not controllable, so how to achieve this?
- Let's look at what engineers do in other fields

When you want to build a race car. . .

. . . adapted to wet tracks

. . . in a dry country . . .



. . . you can simulate it.

But then, you have

- To assess models
- Technical burden
- No real car

Why don't you. . .

just control the climate?

That's Emulation

Emulation in each Science

Studying earthquake effects on bridges

Studying tsunamis

Studying Coriolis effect and stratification vs. viscosity

Studying climate change effects on ecosystems

(who said that science is not fun??)

Emulating Distributed Computer Systems

Possible Approaches in the literature

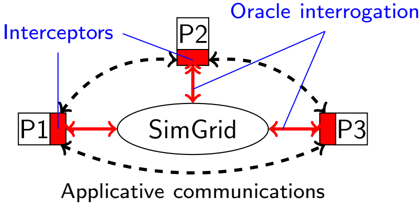
- Performance Degradation: resource burners, usage capping
- Complete Emulation: trick applications into virtual realities

Emulation in EDGE

- WrekAvoc: degradation and capping
- SimTerpose: emulation by mediation through the SimGrid simulator

New results in 2010

- WrekAvoc: Complete rework to handle multi cores properly
- SimTerpose: Several feasibility studies (Java, Unix)



Agenda for 2011

- WrekAvoc: rework the network
- SimTerpose: finish / polish
  Use to study MPI runtimes

Breaking RSA-768 cryptosystems on Grid'5000 (1/2)

Context: Integer factorization problem $\rightsquigarrow$ security of the RSA cryptosystem

vs.

This is no production settings ;)

- We care about feasibility limits, not about people's private data
- Here: given access to a shared resource like a grid, is the task any easier?
- Also, the amount of required resources for factoring matters.

Big picture

- Previous record 663 bits, 2005.
- Used algorithm: Number Field Sieve.
- Core is linear algebra: solve an homogeneous linear system over GF(2)
  - $\approx$ 10 years ago: supercomputer; since 2007: "in-house" HPC cluster
  - 2009: do it on Grid'5000. Prove that we can.

Breaking RSA-768 cryptosystems on Grid'5000 (2/2)

Initial Workplan: split the computation between groups

- EPFL (Lausanne, Switzerland); uses lab cluster; $\leadsto$ 50%
- NTT (Tokyo, Japan); uses lab cluster; $\leadsto$ 10%
- INRIA/CARAMEL (Nancy, France) uses grid platform; $\leadsto$ 40%

Conclusions

- The cryptosystem were broken, resulting in a very large press coverage
- Grid'5000 did 40% of the work; 3 calendar months, 2 months fully working
- First time such a computation (partly) on a grid (for the linear system)
- Proved the viability of a grid setup for this purpose.

- Many rough edges were found and fixed in Grid'5000
- This helped understanding what needs to be improved in our infrastructure

Future: ScaLab (industrializing the experiments)

Layer 3

Experimental methodology: experiment design & planning ; description of scenarios, of experimental conditions ; definition of metrics ; laboratory journal ; analysis and visualization of results

Layer 2

Orchestration of experiments: organize the execution of complex and large-scale experiments ; run experiments unattended and efficiently ; handles failures ; compose experiments

Layer 1

Basic services: common tools required by most experiments

Interact w/ testbed find/reserve/configure resources	**Control a large number of nodes**	**Change experimental conditions**
Check resources before using them	**Manage data**	**Instrument and monitor; extract data**

Layer 0

Experimental testbed (Grid'5000): provides reconfigurable hardware and network, isolation, some instrumentation and monitoring

Conclusion

Computer Science is just like other Sciences

- Experimental facilities are mandatory (even if somehow rigid)
- Emulators are the ultimate scientific instruments (even if very complex)
- Computational Science is extremely powerful (even if tedious to get right)
- All available research methodologies must be combined and leveraged
- Grid'5000 and SimGrid are world leading tools (more to come ;)

	Whiteboard	Simulation	Experimental Facilities	Emulation	Production Platforms
Idea	☺☺				
Algorithm	☺	☺☺			
Prototype		☺	☺☺		
Application			☺☺	☺	
Product					☺

Mutual benefices of collaborations within the community

- Applications become possible, tackle new challenges, strategic advantage
- Fundamental work gets concrete implications, and invaluable feedback
- Our project still a bit young (2010), but getting full gear

EDGE Project in 2011

Experimentation and Distributed systems at Large Scale

1. Experimental methodologies for large scale computer systems
   - Facilities: large scale administration (*new:* graphene cluster – many nodes)
   - Emulation: WrekAvoc (*new:* rewrite) and Simterpose (*new:* feasibility)
   - Simulation: SimGrid project (*new:* SMPI now mature)
   - Overall Organization: Scalable Laboratory (new operation)

2. Reasoned usage of modern computational platforms
   - Applications:
     - Analysis of crypto-systems (*new:* RSA 768 broken)
     - Theorem provers (*new:* testing framework $\rightsquigarrow$ prover comparison/competitions)
     - P2P systems (new: evaluation of a P2P wiki with concurrent editing)
   - Animation of the community:
     - Project bootstraping (first year in CPER)
     - Towards academics (*new:* user day / courses targeting prospective users)
     - Towards industries (*new:* ad hoc intl; Accelor-Mital – in vain so far)
     - Towards production grids (*new:* study EGEE on Grid'5000; contact INRA)

Question slides

Studying Computer Systems

Computers are eminently artificial artifacts

- Humans built them completely, they contain only what we've put in it

$\Rightarrow$ Theoretical (maths) methodology to study it

Computer systems complexity getting tremendous

- Heterogeneity of components (hosts, links)
  - Quantitative: CPU clock, link bandwidth and latency
  - Qualitative: ethernet *vs* myrinet *vs* quadrics; amd64 *vs* ARM *vs* GPU
- Dynamicity
  - Quantitative: resource sharing $\rightsquigarrow$ availability variation
  - Qualitative: resource come and go (churn, failures)
- Complexity
  - Hierarchical systems: grids of clusters of multi-processors being multi-cores
  - Deep software stacks: Middleware, Web Services, mashups
  - Multi-hop nets, high latencies; Interference comput./comm. (disk/memory)

Computer Systems as Natural Objects

- The complexity is so high that we cannot understand them fully anymore
- Frankenstein effect, but allows to use computers to understand computers

Assessing Distributed Applications Correction

- Absence of crash / data corruption (like always)
- Absence of race condition / deadlocks / livelocks (classic in multi-entities)
- Feal with lack of central time and central memory (specific to distributed)

Correction Assessment $\leadsto$ Formal Methods

- Facilities: Experience plans limited, by abilities or by time
- Simulation: How to decide if coverage is sufficient?
- Proof assistants: semi-automated proof demonstration (tedious for users)
- Model checking: Exhaustive state space exploration, search counter examples

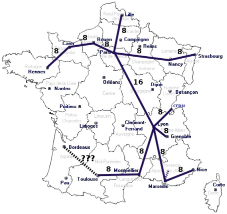
	Experimental Facilities	Emulation	Simulation	Proofs	Model Checking
Performance Assessment	☺☺	☺☺	☺☺	☹☹	☹☹
Experimental Bias	☺☺	☺	☹	(n/a)	(n/a)
Experimental Control	☹☹	☺	☺☺	(n/a)	(n/a)
Ease of Use	☹	☹☹	☺☺	☹☹	☺
Correction Assessment	☹☹	☹	☹	☺☺	☺
Result if failed	(n/a)	(n/a)	(n/a)	☹	☺☺

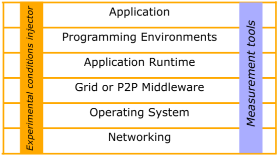
In vivo approach: Direct Experimentation

- Principle: Real applications, Real environment (with reduced external noise)
- Challenges: Not trivial nor immediate. Experimental control? Reproducibility?

Grid'5000 project: world leading scientific instrument for dist. apps

- Instrument for research in computer science (*deploy* your own OS)
- 9 sites, 1500 nodes (3000 cpus, 4000 cores); dedicated 10Gb links





Other existing platforms

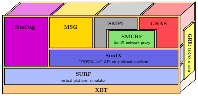
- PlanetLab: No experimental control $\Rightarrow$ no reproducibility
- Production Platforms (EGEE/EGI): must use provided middleware
- FutureGrid: US experimental platform loosely inspired from Grid'5000

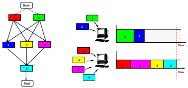
In silico approach: Simulated Experiments

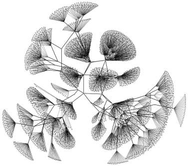
- Principle: Prototypes of applications, models of platforms
- Challenges: Get realistic results (experimental bias)

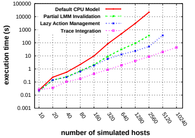
SimGrid: generic simulation framework for distributed applications

- Scalable (time and memory), modular, portable. +70 publications.
- Collaboration Loria / Inria Rhône-Alpes / CCIN2P3 / U. Hawaii





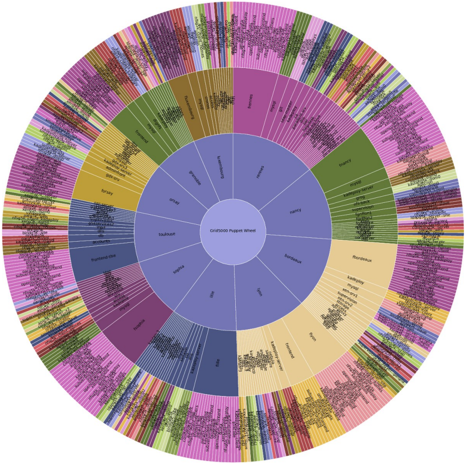




Other existing tools

- *Large* amount of existing simulator for distributed platforms:
  GridSim, ChicSim, GES; P2PSim, PlanetSim, PeerSim; ns-2, GTNetS.
- Few are really usable: Diffusion, Software Quality Assurance, Long-term availability
- No other study the validity, the induced experimental bias

System Administration Challenges



Goals and means

- Automating to factorize
  From 12 to 6 people
- Unique domain
  Intervention range unlimited
- Receipts in a central git
  - Puppet for servers
  - Chef for images
- Capistrano to push configs

Results

- Most know how encoded in receipts
  (young engineers-friendly)
- Hard to handle HPC hosts this way