Experimenting on Architectures for High Performance Computing

Lucas Nussbaum
lucas.nussbaum@loria.fr

École ARCHI 2017
General Outline

1. HPC architectures
2. Experimentation and reproducible research in Computer Science
3. Grid’5000: a Large-Scale Instrument for Parallel and Distributed Computing Experiments
HPC Architectures

Several slides taken from Giovanni Erbacci (CINECA) and Jack Dongarra. Thanks!
Computational Sciences

Computational science (with theory and experimentation), is the “third pillar” of scientific inquiry, enabling researchers to build and test models of complex phenomena.

Quick evolution of innovation:
- Instantaneous communication
- Geographically distributed work
- Increased productivity
- More data everywhere
- Increasing problem complexity
- Innovation happens worldwide
Computational Sciences today

**Multidisciplinary and multiscale problems**

**Coupled applications**

- Full simulation of engineering systems
- Full simulation of biological systems
- Astrophysics
- Materials science
- Bio-informatics, proteomics, pharmaco-genetics
- Scientifically accurate 3D functional models of the human body
- Biodiversity and biocomplexity
- Climate and Atmospheric Research
- Energy
- Digital libraries for science and engineering

**Large amount of data**

**Complex mathematical models**
Moore's Law

- Empirical law which states that the complexity of devices (number of transistors per square inch in microprocessors) doubles every 18 months.
- Gordon Moore, INTEL co-founder, 1965

- It is estimated that Moore's Law will still hold in the near future but applied to the number of cores per processor
Performance Development of HPC over the Last 24 Years from the Top500

My Laptop 70 Gflop/s
My iPhone 4 Gflop/s
6-8 years
Hardware

- Tradeoff between:
  - Performance (FLOPS)
  - Cost
    - Purchase
    - Operation (energy consumption  \(\sim\) cooling)
Intel CPUs (Broadwell-EP, dual-CPU)

(also: Turbo-Boost)

by Gérald Monard (dir. mésocentre EXPLOR)
Communication layer

- Key performance factor: latency
  - Many small, blocking messages during computations

- Specific networking technologies:
  - Myrinet (ancient, 28% of TOP500 in 2005)
  - InfiniBand (main vendor: Mellanox)
  - Intel Omni-Path (since 2015)
  - Support for collective communications in hardware
## 10G-Ethernet vs InfiniBand

RTT between two nodes (using Intel MPI Benchmarks’ PingPong)

### 10G-Ethernet

<table>
<thead>
<tr>
<th>#bytes</th>
<th># repetitions</th>
<th>t [usec]</th>
<th>Mbytes/sec</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1000</td>
<td>17.17</td>
<td>0.00</td>
</tr>
<tr>
<td>1</td>
<td>1000</td>
<td>15.74</td>
<td>0.06</td>
</tr>
<tr>
<td>2</td>
<td>1000</td>
<td>15.06</td>
<td>0.13</td>
</tr>
<tr>
<td>4</td>
<td>1000</td>
<td>15.21</td>
<td>0.25</td>
</tr>
<tr>
<td>8</td>
<td>1000</td>
<td>15.40</td>
<td>0.50</td>
</tr>
<tr>
<td>16</td>
<td>1000</td>
<td>15.10</td>
<td>1.01</td>
</tr>
<tr>
<td>32</td>
<td>1000</td>
<td>15.05</td>
<td>2.03</td>
</tr>
<tr>
<td>64</td>
<td>1000</td>
<td>15.59</td>
<td>3.92</td>
</tr>
<tr>
<td>128</td>
<td>1000</td>
<td>15.72</td>
<td>7.77</td>
</tr>
<tr>
<td>256</td>
<td>1000</td>
<td>18.61</td>
<td>13.12</td>
</tr>
<tr>
<td>512</td>
<td>1000</td>
<td>23.91</td>
<td>20.42</td>
</tr>
<tr>
<td>1024</td>
<td>1000</td>
<td>28.55</td>
<td>34.20</td>
</tr>
<tr>
<td>2048</td>
<td>1000</td>
<td>49.10</td>
<td>39.78</td>
</tr>
<tr>
<td>4096</td>
<td>1000</td>
<td>61.99</td>
<td>63.02</td>
</tr>
<tr>
<td>8192</td>
<td>1000</td>
<td>61.75</td>
<td>126.53</td>
</tr>
<tr>
<td>16384</td>
<td>1000</td>
<td>62.36</td>
<td>250.56</td>
</tr>
<tr>
<td>32768</td>
<td>1000</td>
<td>92.19</td>
<td>338.98</td>
</tr>
<tr>
<td>65536</td>
<td>640</td>
<td>164.89</td>
<td>379.04</td>
</tr>
<tr>
<td>131072</td>
<td>320</td>
<td>298.79</td>
<td>418.35</td>
</tr>
<tr>
<td>262144</td>
<td>160</td>
<td>485.18</td>
<td>515.28</td>
</tr>
<tr>
<td>524288</td>
<td>80</td>
<td>860.18</td>
<td>581.27</td>
</tr>
<tr>
<td>1048576</td>
<td>40</td>
<td>1310.61</td>
<td>763.00</td>
</tr>
<tr>
<td>2097152</td>
<td>20</td>
<td>2428.78</td>
<td>823.46</td>
</tr>
<tr>
<td>4194304</td>
<td>10</td>
<td>4810.80</td>
<td>831.46</td>
</tr>
</tbody>
</table>

### InfiniBand FDR (56 Gb/s)

<table>
<thead>
<tr>
<th>#bytes</th>
<th># repetitions</th>
<th>t [usec]</th>
<th>Mbytes/sec</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1000</td>
<td>1.19</td>
<td>0.00</td>
</tr>
<tr>
<td>1</td>
<td>1000</td>
<td>1.22</td>
<td>0.78</td>
</tr>
<tr>
<td>2</td>
<td>1000</td>
<td>1.23</td>
<td>1.55</td>
</tr>
<tr>
<td>4</td>
<td>1000</td>
<td>1.23</td>
<td>3.10</td>
</tr>
<tr>
<td>8</td>
<td>1000</td>
<td>1.24</td>
<td>6.15</td>
</tr>
<tr>
<td>16</td>
<td>1000</td>
<td>1.25</td>
<td>12.16</td>
</tr>
<tr>
<td>32</td>
<td>1000</td>
<td>1.27</td>
<td>24.07</td>
</tr>
<tr>
<td>64</td>
<td>1000</td>
<td>1.32</td>
<td>46.08</td>
</tr>
<tr>
<td>128</td>
<td>1000</td>
<td>1.96</td>
<td>62.20</td>
</tr>
<tr>
<td>256</td>
<td>1000</td>
<td>2.07</td>
<td>118.17</td>
</tr>
<tr>
<td>512</td>
<td>1000</td>
<td>2.21</td>
<td>221.30</td>
</tr>
<tr>
<td>1024</td>
<td>1000</td>
<td>2.53</td>
<td>386.29</td>
</tr>
<tr>
<td>2048</td>
<td>1000</td>
<td>3.12</td>
<td>625.68</td>
</tr>
<tr>
<td>4096</td>
<td>1000</td>
<td>3.71</td>
<td>1052.75</td>
</tr>
<tr>
<td>8192</td>
<td>1000</td>
<td>5.11</td>
<td>1527.79</td>
</tr>
<tr>
<td>16384</td>
<td>1000</td>
<td>6.74</td>
<td>2319.08</td>
</tr>
<tr>
<td>32768</td>
<td>1000</td>
<td>9.37</td>
<td>3334.06</td>
</tr>
<tr>
<td>65536</td>
<td>640</td>
<td>14.48</td>
<td>4315.46</td>
</tr>
<tr>
<td>131072</td>
<td>320</td>
<td>25.00</td>
<td>4999.99</td>
</tr>
<tr>
<td>262144</td>
<td>160</td>
<td>45.54</td>
<td>5489.57</td>
</tr>
<tr>
<td>524288</td>
<td>80</td>
<td>86.79</td>
<td>5761.11</td>
</tr>
<tr>
<td>1048576</td>
<td>40</td>
<td>169.26</td>
<td>5907.99</td>
</tr>
<tr>
<td>2097152</td>
<td>20</td>
<td>336.05</td>
<td>5951.48</td>
</tr>
<tr>
<td>4194304</td>
<td>10</td>
<td>667.61</td>
<td>5991.54</td>
</tr>
</tbody>
</table>
Accelerators

- **GPGPU** (General-purpose computing on graphics processing units), since ~ 2001
  - E.g. Nvidia Tesla for the HPC market
    - P100: 3500 cores @ 1.3 GHz, 8-10 TFLOPS SP, 4-5 TFLOPS DP

- **Xeon Phi** (Intel MIC architecture – Many Integrated Cores)
  - Knights Corner (2013)
    - PCIe card, 60 cores, 240 threads, 1 - 1.2 TFLOPS
    - Virtualization layer to run standard x86 code
  - Knights Landing (2016)
    - 64-72 cores (x4 threads), 2.6 - 3.4 TFLOPS
    - Host processor (CPU) optionally with integrated Omni-Path adapter
Storage

- Parallel file systems
  - Lustre, GPFS, BeeGFS, etc.

- Local storage on nodes, sometimes

- NVM Express (NVMe) \(\approx\) SSD on PCIe
Outside of the Intel world

- China leading TOP500 with *Sunway TaihuLight System*
  - Using 40,960 custom-made Sunway SW26010 CPUs (1.4 GHz)
  - (Because U.S. banned Intel from supplying Xeon chips to top 4 China supercomputing centers)

- Some IBM Blue Gene/Q (IBM A2 processor, Power architecture)

- Some interest for ARM CPUs
  - Because of higher energy efficiency (e.g. big.LITTLE)
Challenge: efficient use in software

- HPC software: runtime, libraries, applications, tightly coupled with hardware
- A lot of legacy code (often in FORTRAN)
- With a lot of domain-specific knowledge
- Difficult to adjust to new architectures
- Also, speedup limited by the sequential part (Amdahl’s Law)
  - Harder to make use of many slower cores
Experimentation and Reproducible Research in Computer Science
Validation in (Computer) Science

- Two classical approaches for validation:
  - Formal: equations, proofs, etc.
  - Experimental, on a scientific instrument
- Often a mix of both:
  - In Physics
  - In Computer Science
- Quite a lot of formal work in Computer Science
- But also quite a lot of experimental validation
  - Distributed computing, networking → testbeds (IoT-LAB, Grid’5000)
  - Language/image processing → evaluations using large corpuses
Validation in (Computer) Science

- Two classical approaches for validation:
  - Formal: equations, proofs, etc.
  - Experimental, on a scientific instrument

- Often a mix of both:
  - In Physics
  - In Computer Science

- Quite a lot of formal work in Computer Science

- But also quite a lot of experimental validation
  - Distributed computing, networking \(\sim\) testbeds (IoT-LAB, Grid’5000)
  - Language/image processing \(\sim\) evaluations using large corpuses

How good are we at performing experiments?
(Poor) state of experimentation in CS

- 1994: survey of 400 papers\(^1\)
  - among published CS articles in ACM journals, 40%-50% of those that require an experimental validation had none

- 1998: survey of 612 papers\(^2\)
  - too many papers have no experimental validation at all
  - too many papers use an informal (assertion) form of validation

- 2009 update: situation is improving\(^3\)

---


(Poor) state of experimentation in CS (2)

- Most papers do not use even basic statistical tools

<table>
<thead>
<tr>
<th>Year</th>
<th>Tot. papers</th>
<th>With error bars</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>2007</td>
<td>89</td>
<td>5</td>
<td>5.6</td>
</tr>
<tr>
<td>2008</td>
<td>89</td>
<td>3</td>
<td>3.4</td>
</tr>
<tr>
<td>2009</td>
<td>86</td>
<td>2</td>
<td>2.4</td>
</tr>
<tr>
<td>2010</td>
<td>90</td>
<td>6</td>
<td>6.7</td>
</tr>
<tr>
<td>2011</td>
<td>81</td>
<td>7</td>
<td>8.6</td>
</tr>
<tr>
<td>2007-2011</td>
<td>435</td>
<td>23</td>
<td>5.3</td>
</tr>
</tbody>
</table>

- 2007: Survey of simulators used in P2P research
  - Most papers use an unspecified or custom simulator

---

4Study carried out by E. Jeannot.
State of experimentation in other sciences

▶ 2008: Study shows lower fertility for mices exposed to transgenic maize

♦ AFSSA report\(^6\):
  ★ Several calculation errors have been identified
  ★ led to a false statistical analysis and interpretation

\(^6\)Opinion of the French Food Safety Agency (Afssa) on the study by Velimirov et al. entitled “Biological effects of transgenic maize NK603xMON810 fed in long-term reproduction studies in mice”
State of experimentation in other sciences

- 2008: Study shows lower fertility for mices exposed to transgenic maize
  - AFSSA report\(^6\):
    - Several calculation errors have been identified
    - led to a false statistical analysis and interpretation

- 2011: CERN Neutrinos to Gran Sasso project: faster-than-light neutrinos
  - 2012: caused by timing system failure

---

\(^6\)Opinion of the French Food Safety Agency (Afssa) on the study by Velimirov et al. entitled “Biological effects of transgenic maize NK603xMON810 fed in long-term reproduction studies in mice”
State of experimentation in other sciences

- 2008: Study shows lower fertility for mices exposed to transgenic maize
  - AFSSA report\(^6\):
    - Several calculation errors have been identified
    - led to a false statistical analysis and interpretation

- 2011: CERN Neutrinos to Gran Sasso project: faster-than-light neutrinos
  - 2012: caused by timing system failure

- 😞 Not everything is perfect

- 😊 But some errors are properly identified
  - Stronger experimental culture in other (older?) sciences?
    - Long history of costly experiments, scandals, ...
Reproducible Research movement

- Originated mainly in computational sciences (Computational biology, data-intensive physics, etc.)
- Explores methods and tools to enhance experimental practices
  - Enable others to reproduce and build upon one’s work
- Several different motivations
Reproducible Research movement

- Originated mainly in computational sciences (Computational biology, data-intensive physics, etc.)
- Explores methods and tools to enhance experimental practices
  - Enable others to reproduce and build upon one’s work
- Several different motivations
Do The Right Thing™

- Fundamental basis of the scientific method
- K. Poppler, 1934: *non-reproducible single occurrences are of no significance to science*
- Increases transparency, reduces rejection of the scientific community (climate, GMO)
Frustration as a reader or reviewer

This may be an interesting contribution but:

▶ This average value must hide something
Frustration as a reader or reviewer

This may be an interesting contribution but:

- This *average value* must hide something
- As usual, there is no *confidence interval*, I wonder about the variability and whether the difference is *significant* or not
Frustration as a reader or reviewer

This may be an interesting contribution but:

- This **average value** must hide something
- As usual, there is no **confidence interval**, I wonder about the variability and whether the difference is **significant** or not
- That can’t be true, I’m sure they **removed some points**
Frustration as a reader or reviewer

This may be an interesting contribution but:

- This **average value** must hide something
- As usual, there is no **confidence interval**, I wonder about the variability and whether the difference is **significant** or not
- That can’t be true, I’m sure they **removed some points**
- Why is this graph in **logscale**? How would it look like otherwise?
Frustration as a reader or reviewer

This may be an interesting contribution but:

- This average value must hide something
- As usual, there is no confidence interval, I wonder about the variability and whether the difference is significant or not
- That can’t be true, I’m sure they removed some points
- Why is this graph in logscale? How would it look like otherwise?
- The authors decided to show only a subset of the data. I wonder what the rest looks like
Frustration as a reader or reviewer

This may be an interesting contribution but:

- This **average value** must hide something
- As usual, there is no **confidence interval**, I wonder about the variability and whether the difference is **significant** or not
- That can’t be true, I’m sure they **removed some points**
- Why is this graph in **logscale**? How would it look like otherwise?
- The authors decided to show only a **subset of the data**. I wonder what the rest looks like
- There is no label/legend/… **What is the meaning of this graph?** If only I could access the generation script
Frustration as an author

▶ I thought I used the same parameters but I’m getting different results!
Frustration as an author

- I thought I used the same parameters but I’m getting different results!
- The new student wants to compare with the method I proposed last year
Frustration as an author

- I thought I used the same parameters but I’m getting different results!
- The new student wants to compare with the method I proposed last year
- My advisor asked me whether I took care of setting this or this but I can’t remember

Lucas Nussbaum
HPC Architectures
Frustration as an author

- I thought I used the same parameters but I’m getting different results!
- The new student wants to compare with the method I proposed last year
- My advisor asked me whether I took care of setting this or this but I can’t remember
- The damned fourth reviewer asked for a major revision and wants me to change figure 3 :(
Frustration as an author

- I thought I used the same parameters but I’m getting different results!
- The new student wants to compare with the method I proposed last year
- My advisor asked me whether I took care of setting this or this but I can’t remember
- The damned fourth reviewer asked for a major revision and wants me to change figure 3 :(  
- Which code and which data set did I use to generate this figure?
Frustration as an author

- I thought I used the same parameters but I’m getting different results!
- The new student wants to compare with the method I proposed last year.
- My advisor asked me whether I took care of setting this or this but I can’t remember.
- The damned fourth reviewer asked for a major revision and wants me to change figure 3 :(
- Which code and which data set did I use to generate this figure?
- It worked yesterday!
Frustration as an author

- I thought I used the same parameters but I’m getting different results!
- The new student wants to compare with the method I proposed last year
- My advisor asked me whether I took care of setting this or this but I can’t remember
- The damned fourth reviewer asked for a major revision and wants me to change figure 3 :(
- Which code and which data set did I use to generate this figure?
- It worked yesterday!
- 6 months later: why did I do that?
Accelerate your research, increase your impact

- Makes it easier to base on your previous work
- Makes it easier for others to base on your work
  - More visibility, more collaborations
  - More citations

*Sharing Detailed Research Data Is Associated with Increased Citation Rate*\(^7\)

---

\(^7\)Heather A. Piwowar et al. “Sharing Detailed Research Data Is Associated with Increased Citation Rate”. In: *PLoS ONE* 2.3 (Mar. 2007), e308. DOI: 10.1371/journal.pone.0000308. URL: [http://dx.plos.org/10.1371/journal.pone.0000308](http://dx.plos.org/10.1371/journal.pone.0000308).
Because you might be forced to

- NSF policy on the dissemination and sharing of research results
- H2020 Open Research Data Pilot\(^8\) (for 20% of H2020):
  
  1. participating projects are required to deposit the research data described above, preferably into a research data repository. […]
  
  2. as far as possible, projects must then take measures to enable for third parties to access, mine, exploit, reproduce and disseminate (free of charge for any user) this research data.

At the same time, projects should provide information via the chosen repository about tools and instruments at the disposal of the beneficiaries and necessary for validating the results, for instance specialised software or software code, algorithms, analysis protocols, etc. Where possible, they should provide the tools and instruments themselves.

- Nothing at ANR yet?

Different types of experimental reproducibility

- Replications that vary little or not at all with respect to the reference experiment
  
  same method, environment, parameters → same result
  
  ♦ Also called Replicability

- Replications that do vary but still follow the same method as the reference experiment
  
  same method, but different {env., params} → same conclusion
  
  ♦ Example: different testbed

- Replications that use different methods to verify the reference experiment results
  
  different method → same conclusion

---

Reproducibility: what are we talking about?

Reproducibility

Replicability

- Reproduction of the original results using the same tools
- Reproduction using different software, but with access to the original code
- Completely independent reproduction based only on text description, without access to the original code

- Reproduction by the original author on the same machine
- Reproduction by someone in the same lab/using a different machine
- Reproduction by someone in a different lab

Courtesy of Andrew Davison (AMP Workshop on Reproducible research)
The research pipeline

Inspired by Roger D. Peng’s lecture on reproducible research, May 2014
Improved by Arnaud Legrand
The research pipeline

**Author**

- Measured Data
- Analytic Data
- Computational Results
- Figures
- Tables
- Numerical Summaries
- Published Article
- Text

**Nature/System/**...

**Protocol**

(Design of Experiments)

**Scientific Question**

**Reader**

Inspired by Roger D. Peng’s lecture on reproducible research, May 2014

Improved by Arnaud Legrand
Inspired by Roger D. Peng’s lecture on reproducible research, May 2014
Improved by Arnaud Legrand
The research pipeline

Try to keep track of the whole chain = Provenance tracking

Inspired by Roger D. Peng’s lecture on reproducible research, May 2014
Improved by Arnaud Legrand
Reproducible research challenges

- Better descriptions of each step
  - Executable descriptions?
  - Efficient/optimal descriptions?

- Facilitate/automate provenance tracking
  - → move burden away from experimenter
  - Testbeds or experiment management tools with built-in support for provenance collection?

- Ensure that provenance data is sufficient/complete

- Provide sustainable/durable/dependable long-term storage
  - Stable infrastructure
  - Open, standard formats

- Keep stable references between article, code, data
Solutions for reproducible analysis

Note: *Analysis* is generally not very domain-specific
An *Provenance-Rich* Paper: ALPS2.0

The ALPS project release 2.0: Open source software for strongly correlated systems


1Theoretische Physik, EIT Zurich, 8093 Zurich, Switzerland
2Department of Physics, Colorado State University, Fort Collins, CO 80523, USA
3Institut für Theoretische Physik, Technische Universität Graz, A-8010 Graz, Austria
4Department of Physics and Astronomy, University of Wyoming, Laramie, Wyoming 82071, USA
5Scientific Computing and Imaging Institute, University of Utah, Salt Lake City, Utah 84112, USA
6Institut für Theoretische Physik, Georg-August-Universität Göttingen, Göttingen, Germany
7Columbia University, New York, NY 10027, USA
8Bethe Center for Theoretical Physics, Universität Bonn, Nussallee 12, 53115 Bonn, Germany

9Corresponding author: troyer@comp-phys.org

**Figure 3.** In this example we show a data collapse of the Binder Cumulant in the classical Ising model. The data has been produced by remotely run simulations and the critical exponent has been obtained with the help of the VisTrails parameter exploration functionality.

**Figure 4.** The visTrails workflow for the parameter sweep of the classical Ising model.

**Workflow**

- **Libraries**
  - ALPS 2.0
  - matplotlib

**Data**

Simulation Results

VCR: a universal identifier for computational results

Chronicing computations in real-time

VCR computation platform Plugin = Computation recorder

Regular program code

```python
figure1 = plot(x)
save(figure1,'figure1.eps')

> file /home/figure1.eps saved
```

 Courtesy of Matan Gavish and David Donoho (AMP Workshop on Reproducible research)
VCR: a universal identifier for computational results

Chronicing computations in real-time

VCR computation platform Plugin = Computation recorder

Program code with VCR plugin

repository vcr.nature.com
verifiable figure1 = plot(x)

> vcr.nature.com approved:
> access figure1 at https://vcr.nature.com/ffaaffb148d7

Courtesy of Matan Gavish and David Donoho (AMP Workshop on Reproducible research)
VCR: a universal identifier for computational results

Word-processor plugin App

\begin{itemize}
  \item \texttt{LaTeX source} \n  \begin{verbatim}
  \includegraphics{figure1.eps}
  \end{verbatim}
  \item \texttt{LaTeX source with VCR package} \n  \begin{verbatim}
  \includeresult{vcr.thelancet.com/ffaaffb148d7}
  \end{verbatim}
  \item Permanently bind printed graphics to underlying result content
\end{itemize}

Courtesy of Matan Gavish and David Donoho (AMP Workshop on Reproducible research)
VCR: a universal identifier for computational results

Figure 3

Time course of serum stimulation (a) Early passage (E. PD30) or late passage (L. PD89) BJ cultures were held in 0.5% serum for 2 days, then stimulated with 10% FBS. RNA levels from cultures at the indicated time points (Cy5 channel) were compared with the uninduced starting culture (Cy3 channel). Positive values indicate higher expression in induced cells; negative values indicate lower expression in induced cells. Question marks indicate that there was insufficient signal for detection. A complete listing of serum-responsive genes from this analysis is provided in Supplementary material. (b) The serum-responsiveness of select senescence-regulated genes in early passage (PD30) BJ fibroblasts.

Lucas Nussbaum HPC Architectures

Courtesy of Matan Gavish and David Donoho (AMP Workshop on Reproducible research)
Sumatra: an "experiment engine" that helps taking notes

- create new record
- has the code changed?
  - yes
    - code change policy
    - diff
    - store diff
  - no
    - find dependencies
      - get platform information
      - run simulation/analysis
      - record time taken
      - find new files
      - add tags
      - save record

- raise exception
- error

Courtesy of Andrew Davison (AMP Workshop on Reproducible research)
Sumatra: an "experiment engine" that helps taking notes

$ smt comment 20110713-174949 "Eureka! Nobel prize here we come."

Courtesy of Andrew Davison (AMP Workshop on Reproducible research)
Sumatra: an "experiment engine" that helps taking notes

$ smt tag "Figure 6"

Courtesy of Andrew Davison (AMP Workshop on Reproducible research)
Sumatra: an "experiment engine" that helps taking notes

<table>
<thead>
<tr>
<th>Label</th>
<th>Reason</th>
<th>Duration</th>
<th>Processes</th>
<th>Simulator Name</th>
<th>Script Repository</th>
<th>Script Main File</th>
<th>Script Version</th>
<th>Date</th>
<th>Time</th>
<th>Tags</th>
</tr>
</thead>
<tbody>
<tr>
<td>20100709-154755</td>
<td>Eureka! Nobel prize here we come.</td>
<td>0.59 s</td>
<td>Python</td>
<td>2.5.2</td>
<td>/Users/andrew/tmp/SumatraTest main.py</td>
<td>396c2020ca50</td>
<td>09/07/2010</td>
<td>15:42:55</td>
<td></td>
<td></td>
</tr>
<tr>
<td>20100709-154309</td>
<td>‘determine whether the gourd is worth 3 or 4 shekels’</td>
<td>0.59 s</td>
<td>Python</td>
<td>2.5.2</td>
<td>/Users/andrew/tmp/SumatraTest main.py</td>
<td>396c2020ca50</td>
<td>09/07/2010</td>
<td>15:43:09</td>
<td></td>
<td></td>
</tr>
<tr>
<td>haggling</td>
<td>‘apparently, it is worth NaN shekels.’</td>
<td>0.59 s</td>
<td>Python</td>
<td>2.5.2</td>
<td>/Users/andrew/tmp/SumatraTest main.py</td>
<td>396c2020ca50</td>
<td>09/07/2010</td>
<td>15:43:20</td>
<td>foo/bar</td>
<td></td>
</tr>
<tr>
<td>20100709-154338</td>
<td>‘test effect of a smaller time constant’</td>
<td>0.59 s</td>
<td>Python</td>
<td>2.5.2</td>
<td>/Users/andrew/tmp/SumatraTest main.py</td>
<td>396c2020ca50</td>
<td>09/07/2010</td>
<td>15:43:38</td>
<td></td>
<td></td>
</tr>
<tr>
<td>haggling repeat</td>
<td>Repeat experiment haggling</td>
<td>0.58 s</td>
<td>Python</td>
<td>2.5.2</td>
<td>/Users/andrew/tmp/SumatraTest main.py</td>
<td>396c2020ca50</td>
<td>09/07/2010</td>
<td>15:43:47</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Courtesy of Andrew Davison (AMP Workshop on Reproducible research)
Git + Org-mode workflow

- Track link between code, experiments and results using Git branches
- Integrates with Org-mode for literate programming

---

\documentclass[a4paper]{article}
\title{Sweave Example 1}
\author{Friedrich Leisch}
\begin{document}
\maketitle
In this example we embed parts of the examples from the \texttt{kruskal.test} help page into a \LaTeX{} document:

\begin{verbatim}
<<>>=
data(airquality)
library(ctest)
kruskal.test(Ozone ~ Month, data = airquality)
@
\end{verbatim}

which shows that the location parameter of the Ozone distribution varies significantly from month to month. Finally we include a boxplot of the data:

\begin{verbatim}
\begin{center}
<<fig=TRUE,echo=FALSE>>=
boxplot(Ozone ~ Month, data = airquality)
@
\end{center}
\end{verbatim}

Sweave Example 1

Friedrich Leisch

May 21, 2007

In this example we embed parts of the examples from the \texttt{kruskal.test} help page into a \LaTeX{} document:

\begin{verbatim}
> data(airquality)
> library(ctest)
> kruskal.test(Ozone ~ Month, data = airquality)

Kruskal-Wallis rank sum test

data: Ozone by Month
Kruskal-Wallis chi-squared = 29.2666, df = 4, p-value = 6.901e-06
\end{verbatim}

which shows that the location parameter of the Ozone distribution varies significantly from month to month. Finally we include a boxplot of the data:
Solutions for reproducible experiments

Note: *Experiments* is generally quite domain-specific
The Distributed Computing point-of-view

the HARD part

- Measured Data
- Nature/System/...
- Protocol (Design of Experiments)
- Scientific Question

the easy part

- Processing Code
- Analysis Code
- Presentation Code
- Figures
- Tables
- Numerical Summaries
- Published Article
- Text

Analysis/experiment feedback loop

Author

Reader
The Distributed Computing point-of-view

- Rely on large, distributed, hybrid, prototype hardware/software
- Measure execution times (makespans, traces, ...)
- Many parameters, very costly and hard to reproduce

Similar issues in e.g. Wireless Sensor Networks research
Experimental environment management

- How to describe/provide the software environment used?
  
  *I used OpenMPI on Debian ☹*
Experimental environment management

- How to describe/provide the software environment used?
  
  *I used OpenMPI on Debian 😞*

- Obvious solution: virtual machines

  Yes, but:

  ♦ Only provides the final result, not the logic behind each change

  ⟷ easy to forget why/when something was customized

  ♦ No synthetic description: the full image must be provided

  ♦ Cannot really be used as a basis for future experiments

  (≈ object vs source code, preferred form for making modifications)
Creating a package with cde

Timeline

CDE: transparent creation of packages

Executing a package with cde-exec

Timeline

CDE: transparent creation of packages

Creating a package with cde

cd /home/pg/expt/
cde python predict_weather.py

---

CDE: transparent creation of packages

Creating a package with cde

cd /home/pg/expt/
cde python predict_weather.py

CDE: transparent creation of packages

Executing a package with cde-exec

```
cd cde-package/cde-root/home/pg/expt/
cde-exec python predict_weather.py
```
CDE: transparent creation of packages\footnote{Philip J. Guo and Dawson Engler. “CDE: Using System Call Interposition to Automatically Create Portable Software Packages”. In: USENIX ATC. 2011.}

But:

- Does not provide the preferred form for making modifications
- Execution is slower (2\% - 30\%) due to \texttt{ptrace}
Kameleon: reproducible software appliances

- Using recipes (high-level description)
  - Similar to cfengine, Puppet, Chef in the sysadmin world

- Persistent cache to allow re-generation without external resources (Linux distribution mirror) \( \leadsto \) self-contained archive

- Supports LXC, Docker, VirtualBox, qemu, Kadeploy images, etc.

---

12 Cristian Camilo Ruiz Sanabria et al. “Reproducible Software Appliances for Experimentation”. In: TRIDENTCOM’2014.
Improving description and control of experiments

- Legacy way of performing experiments: shell commands
  - 😞 time-consuming
  - 😞 error-prone
  - 😞 details tend to be forgotten over time

- Promising solution: automation of experiments
  ~ Executable description of experiments
Tools for automation of experiments

- Several projects around Grid’5000 (but not specific to Grid’5000):
  - Expo (Cristian Ruiz)
  - Execo (Mathieu Imbert)
  - XPFlow (Tomasz Buchert)

- Others, for other scientific domains:
  - Plush/Gush (PlanetLab)
  - OMF, NEPI (Wireless testbeds)

- Features:
  - Ease scripting of experiments in high-level languages (Ruby, Python)
  - Provide useful and efficient abstractions:
    - Testbed management
    - Local & remote execution of commands
    - Data management
  - Engines for more complex processes

---

Experiment description and execution as a Business Process Workflow

Supports parallel execution of activities, error handling, snapshotting, built-in logging, etc.

soon: automatic provenance collection

---

14 Tomasz Buchert et al. “A workflow-inspired, modular and robust approach to experiments in distributed systems”. In: *CCGRID’2014.*
Other related issues and initiatives
Preserving data and software

- No, your homepage is not a durable storage solution
  - Half-life of URLs in IEEE Computer and CACM: four years\(^{15}\)
  - Y2K crisis: in 1999, 40% of companies had either lost or thrown away the original source code for their systems
  - Code Spaces (Git/SVN project hosting in AWS) hacked: all data lost

- Solutions exist:
  - Articles: ArXiv, HAL
  - Data: Zenodo/OpenAire (CERN, EU-funded), ISAAC (CINES), figshare (Cloud-based)
  - Software: Software Heritage
    (need to consider execution environment, interdependences, software evolution \(\sim\) more complex than books/articles/data)

Online journals, companion websites

- Host code, allow execution (sometimes)
- Example: IPOL Journal – Image Processing On Line

Others: DAE, RunMyCode, etc.

http://www.ipol.im/ (demo)
Evaluation campaigns & challenges

- Evaluate several algorithms against each other, on a given set of inputs
- Events co-hosted with conferences
- Examples in the language/signal processing community:
  - Music Information Retrieval Evaluation Exchange (MIREX)
  - Signal Separation Evaluation Campaign (SiSEC)
  - CHiME Speech Separation and Recognition Challenge
  - Shared Task on Parsing of morphologically-rich languages (SPMRL)
Artifacts evaluation / reproducibility committees

- Authors can submit an archive with the material needed to reproduce their results, and get a "Reproducible" stamp on their paper

---

17 http://www.artifact-eval.org/
19 http://db-reproducibility.seas.harvard.edu/
Artifacts evaluation / reproducibility committees

- Authors can submit an archive with the material needed to reproduce their results, and get a "Reproducible" stamp on their paper

- Questions:
  - How easy is it to use the provided artifact? (Easy to reuse)
  - Does the artifact help to reproduce the results from the paper? (Consistent)
  - What is the percentage of the results that can be reproduced? (Complete)
  - Does the artifact describe and demonstrate how to apply the presented method to a new input? (Well documented)

17http://www.artifact-eval.org/
19http://db-reproducibility.seas.harvard.edu/
Artifacts evaluation / reproducibility committees

- Authors can submit an archive with the material needed to reproduce their results, and get a "Reproducible" stamp on their paper

- Questions:
  - How easy is it to use the provided artifact? (Easy to reuse)
  - Does the artifact help to reproduce the results from the paper? (Consistent)
  - What is the percentage of the results that can be reproduced? (Complete)
  - Does the artifact describe and demonstrate how to apply the presented method to a new input? (Well documented)

- Introduced in several conferences:
  - Compilation, parallel computing\(^{18}\): CGO 2015, PPoPP 2015
  - Databases: SIGMOD 2008\(^{19}\), VLDB 2013

\(^{17}\)http://www.artifact-eval.org/
\(^{19}\)http://db-reproducibility.seas.harvard.edu/
Realis @ COMPAS 2013 and 2014

- COMPAS: Conférence en Parallélisme, Architecture et Système
  - French-speaking, mostly for PhD students

- Realis: test reproducibility of papers submitted to COMPAS
  - Participating authors submit their experimentation description
  - Each author reproduces the experiments from another article
    - Get the identical results, without contacting the authors
    - Evaluate the quality (flexibility, robustness) of the approach

- Most results were reproduced (but none without contacting the authors)

```
Reproduction of the article "Modularise les ordonnanceurs de tâches: une approche structurelle"
Beat Wolf, beat.wolf@hefr.ch

Introduction
Les tests ont été exécutés d’après les instructions données dans l’article soumis à Realis2014. La machine utilisée est la même que celle utilisée pour l’article original, on peut donc s’attendre à des résultats très proches des résultats originaux. Comme suggéré dans l’article soumis à Realis, les 3 figures utilisées dans l’article original ont été reproduites. Les chapitres suivants rentrent plus en détail sur ces 3 figures avec une conclusion à la fin.

Figure 3
La figure 3 montre l’influence des réservoirs sur les performances. L’article mentionne comme conclusion principale de cette figure que les performances sont basses pour 5-15 tâches, moyen pour 20 et 25 avec un pic de performance à 30 tâches. Ce nombre de 30 tâches a été utilisé pour la suite des tests.

Dans l’expérience reproduite on peut retrouver les mêmes valeurs clés mentionnées dans l’article original. Les performances avec 5-15 tâches sont faibles, avec 20-25 moyennes et avec un pic à 30 tâches. Les conclusions de l’article original restent donc valides.

Par contre, comme on peut facilement le voir sur dans le graphe, les performances sont généralement environ 7-8 % plus faibles dans la reproduction que dans l’article original. Les performances avec 30+ tâches sont aussi beaucoup plus irrégulières que dans l’article original. Même que ceci n’influence pas la conclusion tirée de la figure, c’est à dire que le nombre de tâches optimal est de 30, ça reste étonnant, car la même machine a été utilisée pour faire les tests. L’auteur original de l’article a proposé l’hypothèse que cette différence vienne d’une différence de version dans les librairies utilisées, notamment CUDA. Les causes exactes de cette différence n’ont pas pu être identifiées. Il est aussi à noter que pour la reproduction, le script de test a été modifié pour faire 10 itérations au lieu
Illustration 1: Figure 3 dans l’article
Illustration 2: Figure 3 reproduite

Lucas Nussbaum HPC Architectures
```
Conclusions

- Reproducible research
  - A way to improve our daily work, with immediate benefits
  - An opportunity to think about our practices
  - A research field of its own

- Many solutions and tools are now ready for use
Grid’5000:
a Large-Scale Instrument for Parallel and Distributed Computing Experiments
Distributed computing: a peculiar field in CS

- **Performance and scalability** are central to results
  - But depend greatly on the environment (hardware, network, software stack, etc.)
  - Many contributions are about *fighting* the environment
    - Making the most out of limited resources
    - Handling performance imbalance $\leadsto$ load balancing
    - Handling faults $\leadsto$ fault tolerance
    - Hiding complexity $\leadsto$ abstractions: middlewares, runtimes
Distributed computing: a peculiar field in CS

- **Performance and scalability** are central to results
  - But depend greatly on the environment (hardware, network, software stack, etc.)
  - Many contributions are about *fighting* the environment
    - Making the most out of limited resources
    - Handling performance imbalance $\leadsto$ load balancing
    - Handling faults $\leadsto$ fault tolerance
    - Hiding complexity $\leadsto$ abstractions: middlewares, runtimes

- Validation of most contributions require experiments
  - Very little formal validation
  - Even for more theoretical work $\leadsto$ simulation (SimGrid, CloudSim)
Distributed computing: a peculiar field in CS

- Performance and scalability are central to results
  - But depend greatly on the environment (hardware, network, software stack, etc.)
  - Many contributions are about fighting the environment
    - Making the most out of limited resources
    - Handling performance imbalance $\leadsto$ load balancing
    - Handling faults $\leadsto$ fault tolerance
    - Hiding complexity $\leadsto$ abstractions: middlewares, runtimes

- Validation of most contributions require experiments
  - Very little formal validation
  - Even for more theoretical work $\leadsto$ simulation (SimGrid, CloudSim)

- But experimenting is difficult and time-consuming, but often neglected
  - How could we perform better experiments?
  - Very similar to (not computational) biology or physics
What’s an experiment: the research pipeline

Based on figure from Roger D. Peng (Coursera lecture on reproducible research)

Author

the easy part

the HARD part

Processing Code

Analysis Code

Presentation Code

Figures

Tables

Published Article

Published Article

Measured Data

Analytic Data

Computational Results

Numerical Summaries

Text

Analysis/experiment feedback loop

Grid’5000 mission: a large-scale, shared testbed to support high-quality, reproducible experiments

Lucas Nussbaum

HPC Architectures
What’s an experiment: the research pipeline

Grid’5000 mission: a large-scale, shared testbed to support high-quality, reproducible experiments

- **Measured Data**
- **Analytic Data**
- **Computational Results**
- **Figures**
- **Tables**
- **Published Article**
- **Text**

**HARD part**

- **Protocol** (Design of Experiments)

**Easy part**

- **Processing Code**
- **Analysis Code**
- **Presentation Code**
- **Numerical Summaries**

**Analysis/experiment feedback loop**

Based on figure from Roger D. Peng (Coursera lecture on reproducible research)
The Grid’5000 testbed

- One of the world-leading testbeds for distributed computing
  - 8 sites, 30 clusters, 840 nodes, 8490 cores
  - Dedicated 10-Gbps backbone network
  - Various HPC networks and accelerators
  - 550 users and 100 publications per year

A meta-grid, meta-cloud, meta-cluster, meta-data-center:
- Used by CS researchers in HPC / Clouds / Big Data / Networking
- To experiment in a fully controllable and observable environment
- Design goals:
  - Support high-quality, reproducible experiments
  - On a large-scale, shared infrastructure
The Grid’5000 testbed

- One of the world-leading testbeds for distributed computing
  - 8 sites, 30 clusters, 840 nodes, 8490 cores
  - Dedicated 10-Gbps backbone network
  - Various HPC networks and accelerators
  - 550 users and 100 publications per year

- A meta-grid, meta-cloud, meta-cluster, meta-data-center:
  - Used by CS researchers in HPC / Clouds / Big Data / Networking
  - To experiment in a fully controlllable and observable environment

- Design goals:
  - Support high-quality, reproducible experiments
  - On a large-scale, shared infrastructure
Landscape – cloud & experimentation

- **Public cloud infrastructures** (AWS, Azure, Google, etc.)
  - 😞 No information/guarantees on placement, multi-tenancy, real performance

- **Private clouds**: Shared observable infrastructures
  - 😍 Monitoring & measurement
  - 😞 No control over infrastructure settings
  - ~ Ability to understand experiment results

- **On-demand clouds – dedicated observable infrastructures** (BonFIRE)
  - 😍 Limited ability to alter infrastructure

- **Bare-metal as a service, fully reconfigurable infrastructure** (Grid’5000)
  - 😍 Control/alter all layers, including virtualization technology, operating system, networking
Introduction

Discovering resources from their description

Reconfiguring the testbed to meet experimental needs

Monitoring experiments, extracting and analyzing data
Discovering resources from their description

- Describing resources → understand results
  - Covering nodes, network equipment, topology
  - Machine-parsable format (JSON) → scripts
  - Archived (State of testbed 6 months ago?)

```
"processor": {
  "cache_l2": 9393603,
  "cache_ll": null,
  "model": "Intel Xeon",
  "instruction_set": "",
  "other_description": "",
  "version": "X3440",
  "vendor": "Intel",
  "cache_l1": null,
  "cache_l1d": null,
  "clock_speed": 2530000000.0
},
"uid": "graphene-1",
"type": "node",
"architecture": {
  "platform_type": "x86_64",
  "smt_size": 4,
  "smp_size": 1
},
"main_memory": {
  "ram_size": 17179869194,
  "virtual_size": null
},
"storage_devices": [
  {
    "model": "Hitachi HDS72103",
    "size": 298023223875.953,
    "driver": "ahci",
    "interface": "SATA II",
    "rev": "JPF0",
    "device": "sda"
  }
],
```
Discovering resources from their description

▶ Describing resources ⟷ understand results
♦ Covering nodes, network equipment, topology
♦ Machine-parsable format (JSON) ⟷ scripts
♦ Archived (State of testbed 6 months ago?)

▶ Verifying the description
♦ Avoid inaccuracies/errors ⟷ wrong results
♦ Could happen frequently: maintenance, broken hardware (e.g. RAM)
♦ Our solution: g5k-checks
★ Runs at node boot (or manually by users)
★ Acquires info using OHAI, ethtool, etc.
★ Compares with Reference API
Discovering resources from their description

- Describing resources → understand results
  - Covering nodes, network equipment, topology
  - Machine-parsable format (JSON) → scripts
  - Archived *(State of testbed 6 months ago?)*

- Verifying the description
  - Avoid inaccuracies/errors → wrong results
  - Could happen frequently: maintenance, broken hardware (e.g. RAM)
  - Our solution: **g5k-checks**
    - Runs at node boot (or manually by users)
    - Acquires info using OHAI, ethtool, etc.
    - Compares with Reference API

- Selecting resources
  - OAR database filled from Reference API
    - `oarsub -p "wattmeter='YES' and gpu='YES'"`
    - `oarsub -l "cluster='a'/nodes=1+cluster='b' and eth10g='Y'/nodes=2,walltime=2"`
Outline

1. Introduction

2. Discovering resources from their description

3. Reconfiguring the testbed to meet experimental needs

4. Monitoring experiments, extracting and analyzing data
Reconfiguring the testbed

- Typical needs:
  - Install specific software
  - Modify the kernel
  - Run custom distributed middlewares (Cloud, HPC, Grid)
  - Keep a stable (over time) software environment
Reconfiguring the testbed

- Typical needs:
  - Install specific software
  - Modify the kernel
  - Run custom distributed middlewares (Cloud, HPC, Grid)
  - Keep a stable (over time) software environment

- Likely answer on any production facility: you can’t

- Or:
  - Install in $HOME, modules → no root access, handle custom paths
  - Use virtual machines → experimental bias (performance), limitations
  - Containers: kernel is shared → various limitations
Reconfiguring the testbed

- Operating System reconfiguration with Kadeploy:
  - Provides a *Hardware-as-a-Service* cloud infrastructure
  - Enable users to deploy their own software stack & get *root* access
  - Scalable, efficient, reliable and flexible: 200 nodes deployed in ~5 minutes

- Customize networking environment with KaVLAN
  - Protect the testbed from experiments (Grid/Cloud middlewares)
  - Avoid network pollution
  - By reconfiguring VLANS ~ almost no overhead
Creating and sharing Kadeploy images

▶ When doing manual customization:
  ♦ Easy to forget some changes
  ♦ Difficult to describe
  ♦ The full image must be provided
  ♦ Cannot really serve as a basis for future experiments (similar to binary vs source code)

▶ Kameleon: Reproducible generation of software appliances
  ♦ Using recipes (high-level description)
  ♦ Persistent cache to allow re-generation without external resources (Linux distribution mirror) → self-contained archive
  ♦ Supports Kadeploy images, LXC, Docker, VirtualBox, qemu, etc.

http://kameleon.imag.fr/
Changing experimental conditions

- Reconfigure experimental conditions with Distem
  - Introduce heterogeneity in an homogeneous cluster
  - Emulate complex network topologies

http://distem.gforge.inria.fr/
Testing Charm++ load balancing with Distem

No load balancing

- Total run time: 473s
- Average CPU usage: 51%

RefineLB

- Total run time: 443s
- Average CPU usage: 59%

- Every 2 minutes, 1/8 of the nodes are downclocked for 2 minutes
- On the figure, node 0 has been downclocked
- Visible improvement thanks to load balancing
Ensuring consistent hardware configuration

- Many hardware performance settings can have a huge impact
  - BIOS and firmware versions
    - Horror story: older disk firmware version on one node caused 10% performance drop
  - Disks read/write cache
  - CPU P-states, C-states, hyperthreading, turbo-boost
    - On Grid’5000: enabled by default, documentation on how to change them
  - NUMA settings: node interleaving, snoop mode

- Goal: uniform configuration inside clusters; some settings defined testbed-wide

- Regression tests to ensure this
Outline

1. Introduction

2. Discovering resources from their description

3. Reconfiguring the testbed to meet experimental needs

4. Monitoring experiments, extracting and analyzing data
Monitoring experiments

Goal: enable users to understand what happens during their experiment

- System-level probes (usage of CPU, memory, disk, with Ganglia)
- Infrastructure-level probes
  - Network, power consumption
  - Captured at high frequency (≈1 Hz)
  - Live visualization
  - REST API
  - Long-term storage
Conclusions

- Grid’5000: a testbed for high-quality, reproducible research on HPC, Clouds, Big Data and Networking

- With a unique combination of features
  - Description and verification of testbed
  - Reconfiguration (hardware, network)
  - Monitoring
  - Support for automation of experiments

- Try it yourself!
  - Free account through the Open Access program
    http://www.grid5000.fr/open-access

Bibliography

- **Resources management:** Resources Description, Selection, Reservation and Verification on a Large-scale Testbed. [http://hal.inria.fr/hal-00965708](http://hal.inria.fr/hal-00965708)

- **Kadeploy:** Kadeploy3: Efficient and Scalable Operating System Provisioning for Clusters. [http://hal.inria.fr/hal-00909111](http://hal.inria.fr/hal-00909111)

- **KaVLAN, Virtualization, Clouds deployment:**
  - Adding Virtualization Capabilities to the Grid'5000 testbed. [http://hal.inria.fr/hal-00946971](http://hal.inria.fr/hal-00946971)
  - Enabling Large-Scale Testing of IaaS Cloud Platforms on the Grid'5000 Testbed. [http://hal.inria.fr/hal-00907888](http://hal.inria.fr/hal-00907888)

- **Kameleon:** Reproducible Software Appliances for Experimentation. [https://hal.inria.fr/hal-01064825](https://hal.inria.fr/hal-01064825)

- **Distem:** Design and Evaluation of a Virtual Experimental Environment for Distributed Systems. [https://hal.inria.fr/hal-00724308](https://hal.inria.fr/hal-00724308)

- **XP management tools:**
  - A survey of general-purpose experiment management tools for distributed systems. [https://hal.inria.fr/hal-01087519](https://hal.inria.fr/hal-01087519)
  - **XPFlow:** A workflow-inspired, modular and robust approach to experiments in distributed systems. [https://hal.inria.fr/hal-00909347](https://hal.inria.fr/hal-00909347)
  - Using the **EXECO** toolbox to perform automatic and reproducible cloud experiments. [https://hal.inria.fr/hal-00861886](https://hal.inria.fr/hal-00861886)
  - **Expo:** Managing Large Scale Experiments in Distributed Testbeds. [https://hal.inria.fr/hal-00953123](https://hal.inria.fr/hal-00953123)

- **Kwapi:** A Unified Monitoring Framework for Energy Consumption and Network Traffic. [https://hal.inria.fr/hal-01167915](https://hal.inria.fr/hal-01167915)

- **Realis’2014:** Reproductibilité expérimentale pour l’informatique en parallélisme, architecture et système. [https://hal.inria.fr/hal-01011401](https://hal.inria.fr/hal-01011401)