# Ultra Scalable Simulation with SimGrid
# USS SimGrid (ANR 08 SEGI 022)

http://uss-simgrid.gforge.inria.fr

*Coordinated by* Martin Quinson (Nancy University)

Paris, September 17 2010

# Context: Large-Scale Distributed Systems

## Cloud Computing

- Large infrastructures underlying the commercial Internet
- Examples: eBay, Amazon, Facebook, Google, ViaMichelin, Voyages-SNCF, etc.
- Main issue: keep up with the load, even when facing flash crowd effects

## Peer-to-peer Systems (P2P)

- Goal: Exploit resources at network edges (storage, CPU, human presence)
- Approach: Decentralized Systems (not Clients/Server; each node does both)
- Promises: Organic growth, infrastructure independence, scalability, robustness
- Issues: Nodes' intermittent connectivity (churn); Network locality; Anonymity

## These systems are in use today, but badly understood

- They deserve a thorough scientific analysis

# How to perform this study

## Classical approaches in science and engineering

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

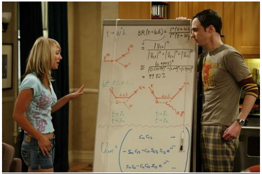
## That's not always desirable (or even possible)

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

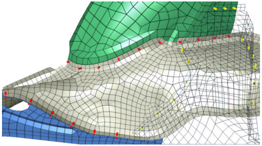
## The third scientific way: *Computational Science*

3. Study in silico using computers
   Modeling / Simulation of the phenomenon or data-mining

⇝ High Performance Computing Systems


The Big Bang Theory


Large Hadron Collider


Car Mesh

# How to perform this study

## Classical approaches in science and engineering

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

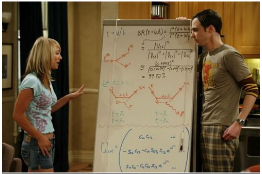
## That's not always desirable (or even possible)

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

## The third scientific way: *Computational Science*

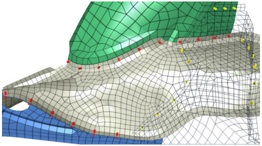
3. Study in silico using computers
   Modeling / Simulation of the phenomenon or data-mining

⇝ High Performance Computing Systems



The Big Bang Theory

Large Hadron Collider



Car Mesh

## These systems deserve very advanced analysis

- Debugging and tuning technically difficult; Induce methodological challenges
- Science of the *in silico science*
- Same benefits for our study as for other sciences

# Large-Scale Distributed Systems Science?

## Requirements for a Scientific Approach

- Reproducible results
  - You can read a paper, reproduce a subset of its results and improve
- Standard tools and methodologies
  - Grad students can learn their use and become operational quickly
  - Experimental scenario can be compared accurately

## Current practice in the field: quite different

- Very little common methodologies and tools, large load of (ad-hoc) tools
  - GridSim, ChicSim, GES; P2PSim, PlanetSim, PeerSim; ns-2, GTNetS
    From 141 P2P sim.papers: 30% custom tool, 50% don't report tool [Naicken06]
  - Few are really usable: Diffusion, Software Quality Assurance, Long-term availability
  - Most rely on straightforward models with no validity assessment
- Experimental settings rarely detailed enough in literature

# Large-Scale Distributed Systems Science?

## Requirements for a Scientific Approach

- Reproducible results
  - You can read a paper, reproduce a subset of its results and improve
- Standard tools and methodologies
  - Grad students can learn their use and become operational quickly
  - Experimental scenario can be compared accurately

## Current practice in the field: quite different

- Very little common methodologies and tools, large load of (ad-hoc) tools
  - GridSim, ChicSim, GES; P2PSim, PlanetSim, PeerSim; ns-2, GTNetS
    From 141 P2P sim.papers: 30% custom tool, 50% don't report tool [Naicken06]
  - Few are really usable: Diffusion, Software Quality Assurance, Long-term availability
  - Most rely on straightforward models with no validity assessment
- Experimental settings rarely detailed enough in literature

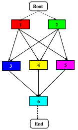
## Purpose of the SimGrid Project

- Allow a scientific approach of Large-Scale Distributed Systems simulation
- Propose ready to use tools enforcing methodological best practices

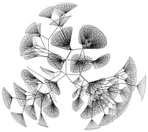
# Simulating Distributed Systems

## Principle

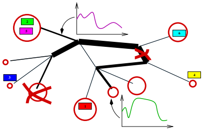
Idea to test



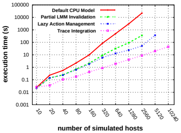

\+ System Model



\+ Experimental setup



⇒ Scientific results




## Advantages

- Less simplistic than proposed theoretical models (which are useful too)
- Better XP control ($\leadsto$ reproducible) than production systems (+ not disruptive)
- Not as tedious, time/labor consuming than experimental platforms
- Plus: Lower technical burden; Quick and easy experiments; What if analysis

## Main challenges

- Validity: Get realistic results (controlled experimental bias)
- Scalability: Simulate *fast enough* problems *big enough*
- Usability: Associated Tools; Ease of use; Applicability to context of interest

# The USS-SimGrid project

## Make SimGrid usable in studies mandating extreme scaling

- Perimeter increase from Grid Computing to Peer-to-peer
- Improving simulation scalability: mandatory but not enough
- Campaign data management pre- & post-processing not trivial anymore

# The USS-SimGrid project

## Make SimGrid usable in studies mandating extreme scaling

- Perimeter increase from Grid Computing to Peer-to-peer
- Improving simulation scalability: mandatory but not enough
- Campaign data management pre- & post-processing not trivial anymore

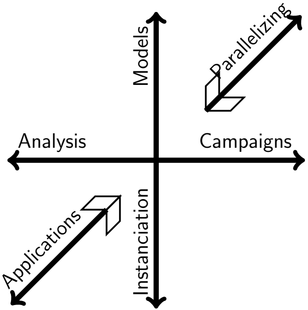
## Project organization

Axis 1 Models
- WP1: New models + validity
- WP2: Model instantiation (usability)

Axis 2 Associated tools
- WP3: Simulation analysis
- WP4: Campaign management

Axis 3 Extreme scalability
- WP5: Parallel and distributed simulation

Axis 4 Transfer, dissemination
- WP6: Applications



# The USS-SimGrid project

## Make SimGrid usable in studies mandating extreme scaling

- Perimeter increase from Grid Computing to Peer-to-peer
- Improving simulation scalability: mandatory but not enough
- Campaign data management pre- & post-processing not trivial anymore

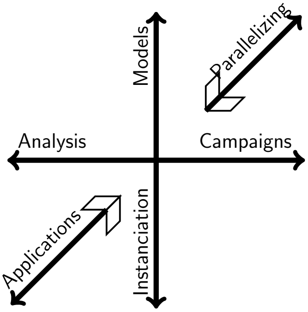
## Project organization

Axis 1 Models
- WP1: New models + validity
- WP2: Model instantiation (usability)

Axis 2 Associated tools
- WP3: Simulation analysis
- WP4: Campaign management

Axis 3 Extreme scalability
- WP5: Parallel and distributed simulation

Axis 4 Transfer, dissemination
- WP6: Applications




**Coming next:** Some scientific achievements on Scalability, Validity and Usability

For each: Challenge; Focus on one result; Envisioned work within the project

# SimGrid Internals in a Nutshell

## Example of user code to execute

Alice

```
Send "toto" to Bob
Listen from Bob
```

Bob

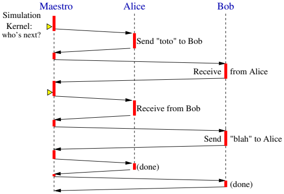
```
Listen from Alice
Send "blah" to Alice
```

# SimGrid Internals in a Nutshell

## Example of user code to execute

Alice
```
Send "toto" to Bob
Listen from Bob
```

Bob
```
Listen from Alice
Send "blah" to Alice
```

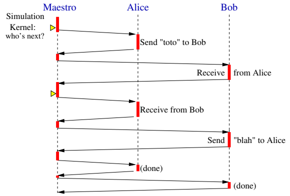


# SimGrid Internals in a Nutshell

## Example of user code to execute

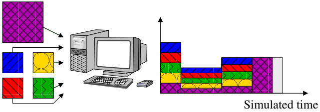
```
Alice
Send "toto" to Bob
Listen from Bob
```

```
Bob
Listen from Alice
Send "blah" to Alice
```




## SimGrid internal Main Loop

1. Run every ready user process in row
   - Each wants to consume resources
   - Assign actions on resources
2. Compute share for actions
3. Get earliest finishing action
4. Unlock user code waiting on this action

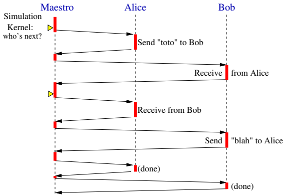


# SimGrid Internals in a Nutshell

## Example of user code to execute

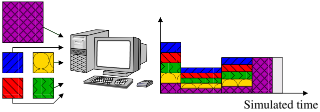
**Alice**
```
Send "toto" to Bob
Listen from Bob
```

**Bob**
```
Listen from Alice
Send "blah" to Alice
```



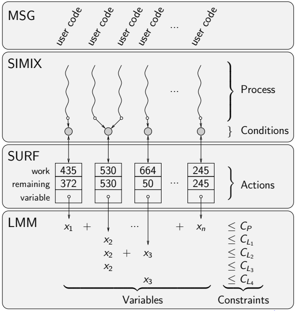

## SimGrid internal Main Loop

1. Run every ready user process in row
   - Each wants to consume resources
   - Assign actions on resources
2. Compute share for actions
3. Get earliest finishing action
4. Unlock user code waiting on this action




## SimGrid Functional Organization

- MSG: User-friendly syntaxic sugar
- Simix: Processes, synchro (SimPosix)
- SURF: Resources usage interface
- Models: Action completion computation



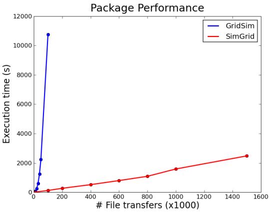
# Agenda

- Introduction, Context and Motivation
- Scientific Achievements
  - Scalability Challenge
  - Validity Challenge
  - Usability Challenge
- Organizational Aspects
  - Work Organization
  - Current Situation
  - Project Outcomes
- Conclusion and Open questions

# Scalability Challenge

## Situation before the project

- Timings from CERN guys

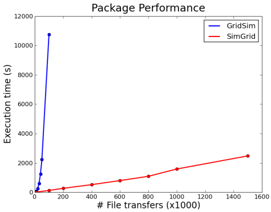



- Maximal amount of user processes
  - GridSim: 10,922 (hard limit)
  - SimGrid: 200k (memory limit, 4Gb)

# Scalability Challenge

## Situation before the project

- Timings from CERN guys

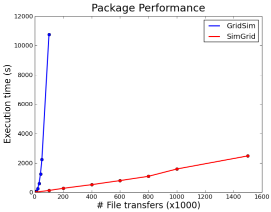



- Maximal amount of user processes
  - GridSim: 10,922 (hard limit)
  - SimGrid: 200k (memory limit, 4Gb)
- But needs of the users:
  - CERN: 300 × bigger than that (10 days/run)
  - BOINC: 600k volatile hosts over a year
- PeerSim simulates millions of processes
  - but with simplistic models only

# Scalability Challenge

## Situation before the project

- Timings from CERN guys




- Maximal amount of user processes
  - GridSim: 10,922 (hard limit)
  - SimGrid: 200k (memory limit, 4Gb)
- But needs of the users:
  - CERN: 300 × bigger than that (10 days/run)
  - BOINC: 600k volatile hosts over a year
- PeerSim simulates millions of processes
  - but with simplistic models only

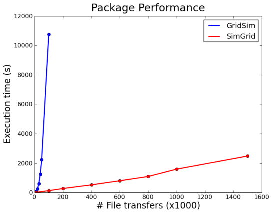
## Scalability constitutes the main objective of the USS SimGrid

- Two aspects: Big enough (large platforms) ⊕ Fast enough (large workload)
- Possible approaches:
  - Algorithmic optimizations: Compact routing representation ⊕ Lazy evaluation
  - Multiple computers: *Distribution* ⊕ *Parallelism*
  - Simpler models (but potential loss of realism)
- USS SimGrid leverages all these approaches

# Scalability Challenge

## Situation before the project

- Timings from CERN guys




- Maximal amount of user processes
  - GridSim: 10,922 (hard limit)
  - SimGrid: 200k (memory limit, 4Gb)
- But needs of the users:
  - CERN: 300 $\times$ bigger than that (10 days/run)
  - BOINC: 600k volatile hosts over a year
- PeerSim simulates millions of processes
  - but with simplistic models only

## Scalability constitutes the main objective of the USS SimGrid

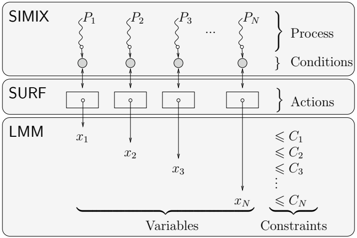
- Two aspects: Big enough (large platforms) $\oplus$ Fast enough (large workload)
- Possible approaches:
  - Algorithmic optimizations: Compact routing representation $\oplus$ Lazy evaluation
  - Multiple computers: *Distribution* $\oplus$ *Parallelism*
  - Simpler models (but potential loss of realism)
- USS SimGrid leverages all these approaches
- Coming now: focus on 2 points

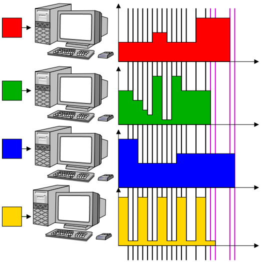
# Simulation Speed Improvement (1/2)



## Context: Volunteer Computing

- One task per CPU; Availability trace; network not relevant to the study



## Lazy Evaluation

- LMM model is a MaxMin system
- Used to recompute it all on each change
- Waste of time if system is loosely coupled
- Ex: 3h to simulate 2500 hosts for one week
  No coupling $\rightsquigarrow$ dumb full recomputes

$\rightsquigarrow$ Invalidate only changed parts of the system

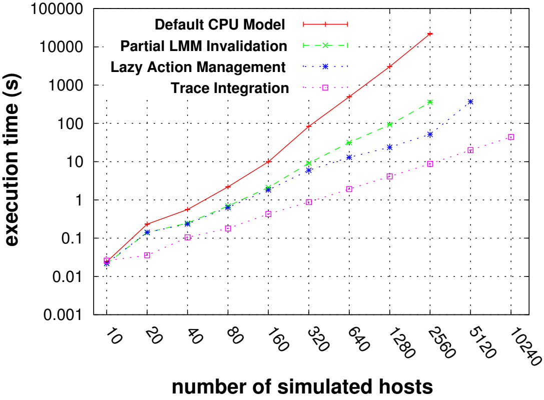
## Availability Trace Integration

- Before: ∀step, ∀action, compute if done
- Waste of time if only one action per resource

$\rightsquigarrow$ Precompute termination date only once

# Simulation Speed Improvement (2/2)





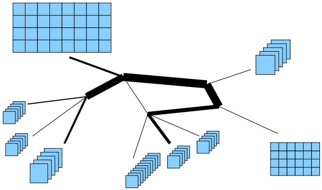

## Results

- From 3 hours to 10 seconds to simulate one week of 2500 dynamic hosts
- Arbitrary speedup depending on scenario (less coupling $\rightsquigarrow$ more speedup)
- Huge gain in typical P2P and Desktop Grid settings
  - 60 times faster than BOINC client simulator
  - 20-30 times faster than SimBA (an ad hoc BOINC simulator designed to scale)

# Memory Scalability: Simpler Models (1/3)



## Classical Network Model in SimGrid



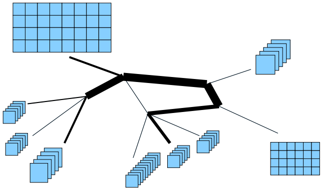
- Precise platform graph
- Needs complete routing table: quadratic size
- Limiting factor to consider larger platforms
- Acquisition/Generation is a problem

- P2P community: constant time for all coms
  Not enough info available to instanciate this

# Memory Scalability: Simpler Models (1/3)



## Classical Network Model in SimGrid



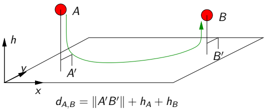
- Precise platform graph
- Needs complete routing table: quadratic size
- Limiting factor to consider larger platforms
- Acquisition/Generation is a problem

- P2P community: constant time for all coms
  Not enough info available to instanciate this

## Simpler models: compact distance labeling

- Assign a label (eg coordinates) to each host
- Evaluate distance between 2 hosts from their labels
- Complexity: linear size, constant time
- Good compact representation for latencies

Ex.: Vivaldi model



# Memory Scalability: Simpler Models (2/3)

Bordeaux, Saclay

## Example of application: Peer-assisted video streaming

- Send a large message to a large number of hosts
- Peers may help by forwarding the message to other peers
- Algorithmic problem: organizing communications to maximize throughput
- Natural value of interest: available bandwidth

## Last-mile model

- Hosts are characterized by their incoming and outgoing bandwidth
- $BW_{A,B} = \min(b_A^{\text{out}}, b_B^{\text{in}})$
- Allows to model the asymmetry of actual bandwidth measures
- Instanciation is possible from a small number of measurements
- Theoretical result: near-optimal allocation for streaming with bounded degree

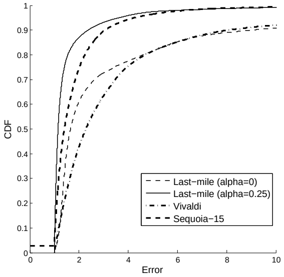
# Memory Scalability: Simpler Models (3/3)

## Precision of the simple models

- Assess quality of recomputed values wrt original
- Comparison made from measures from PlanetLab

$\rightsquigarrow$ *Error* $_{last\_mile} < 2$ for 85% of measurements

- Simple models can provide interesting results
- Asymmetry is an important feature




## Future directions

- Evaluate validity through the behavior of applications
- Combine bandwidth and latency
- Add complexity to the last-mile model for increased precision

# Scalability: Planned Work

## Hierarchical routing: memory footprint (large platforms)

- The current representation relies on a full $N \times N$ routing table
  This table alone exhausts gigabytes for 1000 hosts only
- Exploit hierarchy and regularity to gain several orders of magnitude

## Distribution and Parallelization (large amount of processes)

- Tweaking stack size enable to reach 200,000 user threads (not always possible)
- Adopt a real OS-like architecture to distribute user code on several machines
- Factorize common parts of simulations
- Exploit semantic independence of events to increase parallelism

## Storage modeling

- Modeling the performance of a single hard drive seems impossible
- Stochastic modeling of thousands of tapes and hard drives may be easier
  (in collaboration with the CERN team in charge of the data management)

# Validity Challenge

Context: Models in most simulators are either simplistic, wrong or not assessed

- PeerSim: discrete time, application as automaton; GridSim: naive packet level
- OptorSim, GroudSim: documented as being wrong on heterogeneous platforms

## Quality Levels of Validity

- Level -1: not validated (probably plainly wrong)
- Level 0 (visually ok): a few curves that look similar (generally hides a lot)
- Level 1 (ratios ok): $A < B$ in Simulation $\Leftrightarrow$ $A < B$ in Reality
- Level 2 (prediction abilities): bounded distance between simulation and reality
- Orthogonal to this: need to assess when the model is valid (validity domain)
- Validity evaluation: tricky, requires meticulous attention & sound methodology

# Validity Challenge

Context: Models in most simulators are either simplistic, wrong or not assessed

- PeerSim: discrete time, application as automaton; GridSim: naive packet level
- OptorSim, GroudSim: documented as being wrong on heterogeneous platforms

## Quality Levels of Validity

- Level -1: not validated (probably plainly wrong)
- Level 0 (visually ok): a few curves that look similar (generally hides a lot)
- Level 1 (ratios ok): $A < B$ in Simulation $\Leftrightarrow$ $A < B$ in Reality
- Level 2 (prediction abilities): bounded distance between simulation and reality
- Orthogonal to this: need to assess when the model is valid (validity domain)
- Validity evaluation: tricky, requires meticulous attention & sound methodology

## SimGrid validity before USS: Research focus in SimGrid since 2002

Setting: *Synthetic* App. + *Synthetic* WAN; Compare against packet-level simulator

- Error in percents if: TCP steady state (flows > 10Mb), latency-bound (WAN)
- Wrong estimations when capacity-bound (suspect: max-min sharing)

# Validation Improvements in USS

Nancy, Villeurbanne, Grenoble

First Step: *Synthetic* App. + *Synthetic* WAN. Compare against *GTNetS*

- Some errors were hunted down + unexpected phenomenon were understood

$\leadsto$ The model and its instanciation were considerably improved
Widen validity range to flows > 100Kb and WAN with small latencies

- Sharing mechanism from theoretical literature experimentally proved wrong

# Validation Improvements in USS

Nancy, Villeurbanne, Grenoble

## First Step: *Synthetic* App. + *Synthetic* WAN. Compare against *GTNetS*

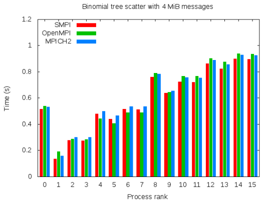
- Some errors were hunted down + unexpected phenomenon were understood

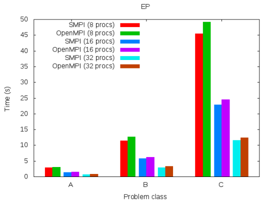
$\leadsto$ The model and its instanciation were considerably improved
Widen validity range to flows > 100Kb and WAN with small latencies

- Sharing mechanism from theoretical literature experimentally proved wrong

## Going Further: developed SMPI $\leadsto$ Real App. (NAS PB) + clusters (LAN)

- Good prediction for short messages is crucial (piecewise linear)
- Need to accurately implement/model collective operation algorithms
- Evaluating weight of computation phases is tricky, numerical instabilities deadly
- Need to account for MPI overhead; what is Real with several MPI implems?




# Validity: Planned Work

## Most of WP1/WP2 done

- Validity is an endless quest and we will pursue our effort on SimGrid validation...
- ...but probably not within the USS project
- Current validity is good enough for P2P and Volunteer Computing settings
- SMPI almost good enough for cluster dimensioning (WP6.1)

## Other efforts: New kind of models (FYI)

- Storage elements (collaboration with CERN)
- Multi-core (taking into account memory consumption)
- Stochastic models for availability/unavailability traces (if users need it)

# Usability Challenge

## Workflow to any Experiments through Simulation

1. Prepare the experimental scenarios (platform, background load, ...)
2. Launch thousands of simulations
3. Post-processing and result analysis

$\rightsquigarrow$ Each simulation is only a brick, we must provide more tools

## Situation before the project

- Others simulators come with *ad hoc* tools (but many *demowares*)
- SimGrid: nothing public/generic, but each user grow home-made scripts

## USS-SimGrid Proposal

1. Workload generation:
   - Platforms (Simulacrum, PDA, MintCAR, ...)
   - Applicative Workload (trace collection+replay)
   - Background Workload
2. Campaign management
3. Single simulation analysis: Visualization

# Visualization Challenges

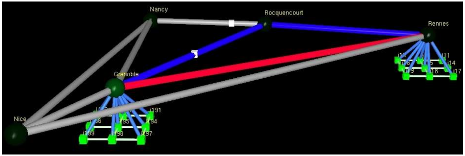
## Simulations can produce a *lot* of logs (even more at large scale)

- Everyone produces ad-hoc parsing scripts
- Not always easy, graphically visualizing more appealing
- Visual inspection to check the correctness of the simulation is crucial

## Building a *demoware* is easy. Helping understanding is harder

- Most of the time *ad hoc*: developed specifically for one simulator/library
- Do not show the right informations: platform/application state, tracing/profiling
- Cumbersome, nearly impossible to adapt: shows what its author wanted to see
- Scale badly

```
$ ./my_simulator | MSG_visualization/colorize.pl
[     0.000][  Tremblay:master    ] Got 3 workers and 6 tasks to process
[     0.000][  Tremblay:master    ] Sending 'Task_0' to 'worker-0'
[     0.148][  Tremblay:master    ] Sending 'Task_1' to 'worker-1'
[     0.148][   Jupiter:worker    ] Processing 'Task_0'
[     0.347][  Tremblay:master    ] Sending 'Task_2' to 'worker-2'
[     0.347][    Fafard:worker    ] Processing 'Task_1'
[     0.476][  Tremblay:master    ] Sending 'Task_3' to 'worker-0'
[     0.476][   Ginette:worker    ] Processing 'Task_2'
[     0.803][   Jupiter:worker    ] 'Task_0' done
[     0.951][  Tremblay:master    ] Sending 'Task_4' to 'worker-1'
[     0.951][   Jupiter:worker    ] Processing 'Task_3'
[     1.003][    Fafard:worker    ] 'Task_1' done
[     1.202][  Tremblay:master    ] Sending 'Task_5' to 'worker-2'
[     1.202][    Fafard:worker    ] Processing 'Task_4'
[     1.507][   Ginette:worker    ] 'Task_2' done
[     1.606][   Jupiter:worker    ] 'Task_3' done
[     1.635][  Tremblay:master    ] All tasks dispatched. Let's stop workers.
[     1.635][   Ginette:worker    ] Processing 'Task_5'
[     1.637][   Jupiter:worker    ] I'm done. See you!
[     1.857][    Fafard:worker    ] 'Task_4' done
[     1.859][    Fafard:worker    ] I'm done. See you!
[     2.666][   Ginette:worker    ] 'Task_5' done
[     2.668][  Tremblay:master    ] Goodbye now!
[     2.668][   Ginette:worker    ] I'm done. See you!
[     2.668][                     ] Simulation time 2.66766
```

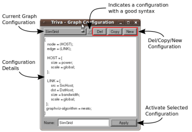


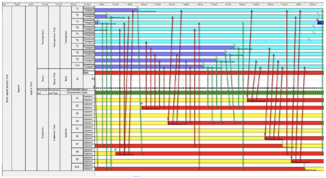
# Visualizing SimGrid Simulations

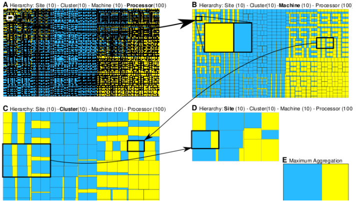


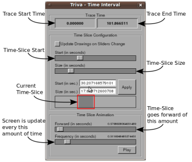
## Triva and Paje: separate (established) projects (L. Schnorr, B. Stein)

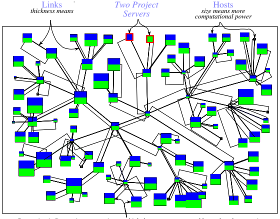
- Generic and dedicated to visualization: SimGrid only produces adapted traces
- Display the right information: intermediate between monitoring and profiling
- Easy navigation in space and time: selection, aggregation, animation
- Scalable: efficient representation and implementation, allows efficient browsing



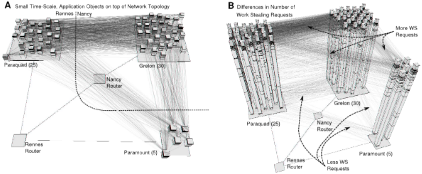






Separation indicates time proportion at which the resource was used by each task categories



# Use case driven research

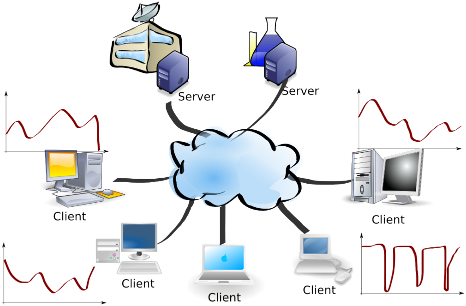
## Detecting Anomalies

- Used to study scheduling issues related to BOINC
- Used to track differences in validity study

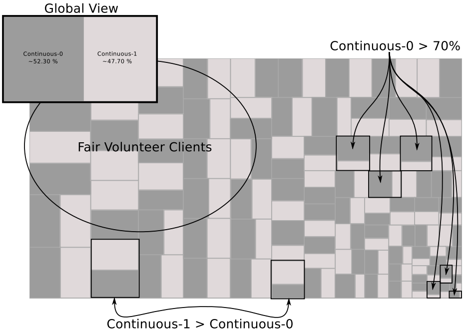
Most of the time, we set up the visualization to *check* something

Often, we *noticed* something else

### Fairness of BOINC




### Bug identification



# Usability: Planned Work

## Experimental Setup Generation

- Adapt to future hierarchical platform description
- Programmative rather than descriptive-only approach
  (from XML files to lua console)

## Campaign Management

- Running and managing results on a grid requires an efficient framework
- Design of Experiments: Methodology to decide *which* and *how many* experiments to do to gain as much insight as possible

## Visualization

- Use case driven research
- Spatial aggregation for graph representations
- Trace comparison: understand what changed from one run to another

# Agenda

- Introduction, Context and Motivation
- Scientific Achievements
  - Scalability Challenge
  - Validity Challenge
  - Usability Challenge
- Organizational Aspects
  - Work Organization
  - Current Situation
  - Project Outcomes
- Conclusion and Open questions

# Collaborative Work

These informations are available in the Mobility livrable, and on the web site

## 11 Visits (2-3 sites; 3-6 people)

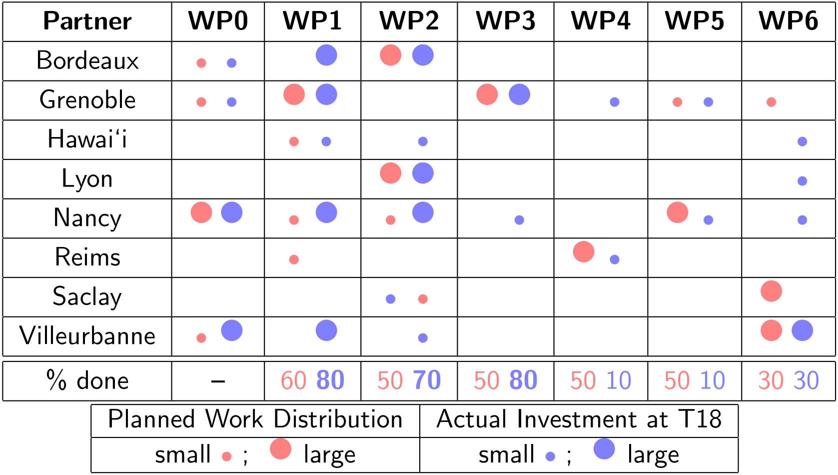
| | Bordeaux | Grenoble | Hawai'i | Lyon | Nancy | Reims | Saclay | Villeurbanne |
|---|---|---|---|---|---|---|---|---|
| Bordeaux | – | | | X | | | | |
| Grenoble | | – | | X | XXX | X | | XX |
| Hawai'i | | | – | X | X | | | |
| Lyon | X | X | X | – | XX | | | – |
| Nancy | | XXX | X | XX | – | | | X |
| Reims | | X | | | | – | | |
| Saclay | | | | | | | – | |
| Villeurbanne | | XX | | – | X | | | – |

(plus monthly audio meetings and daily Instant Messaging)

## 6 Plenary Meetings (all sites; 20+ people)

- January 15-16, 2009: Kickoff meeting
- March 12, 2009: WP4 kickoff
- Jul 9, 2009: T6 meeting
- Dec 8, 2009: T12 meeting
- Apr 12, 2010: T16 meeting
- Sep 6-7, 2010: Evaluation preparation meeting (leaders only)

# Work (re)Distribution




- Initially each partner had its WP of major interest
- In pratice the forces have gathered on WP1, WP2, and WP3
  - The same is likely to happen for WP4 and WP5 in the next 18 months

# Reorganizations and Difficulties

## Consortium Modifications

- At T0: F. Suter moved from Nancy to Villeurbanne (new partner)
  - 1-Year engineer funding moved too
- At T18: F. Le Fessant (Saclay) leaved the project (and his researcher position)
  - Replaced by a new partner: O. Dalle (Nice)

## Scientific Focus Adjustments

- Tremendous improvements on scalability thanks to the force gathering on WP1
  - Parallelization (WP5) for scalability is less urgent
  - ⇝ Refocus on the last subtask of WP5 (Simulation of Distributed Forks)
  - The remaining currently starting

## Hiring Issue

- Reims found a good candidate only at T15 (while WP4 needed it at T0)
- Grenoble found a good candidate for WP4 at T18
  - ⇝ WP4 is now shared between Reims and Grenoble

# Hiring and Financial Status

## ANR funded positions

- Almost all positions have been taken (but 2 postdocs in Bordeaux and Nice)

## *Non-ANR* funded positions

- Engineers: 2 × 2-year positions funded by INRIA ADT program
- PhDs: 2 positions (INRIA and INRIA/Région) related to WP6 and WP2
- Interns: 3 international (WP1) and 1 engineering school (WP2), all INRIA

## Financial Status

| | Spent budget | Taken Positions | |
|---|---|---|---|
| Bordeaux | 22% | 12/24 months | |
| Grenoble | 29% | 36/36 months | |
| Lyon | 81% | 12/12 months | contract ended at T12 |
| Nancy | 42% | 60/60 months | |
| Reims | 11% | 24/24 months | |
| Saclay/Nice | 0.2% | 0/12 months | |
| Villeurbanne | 59% | 9/12 months | convert salaries → dissemination |

Note: aggreed at T0 that T0=15 jan 2009, but officially 15 dec 2008. How to fix?

# Production and Dissemination

## Publications

- 11 international publications (including 2 multi-site publications)
  - SIMUTools (2009 and 2010), IPDPS'10, ICDS'09, CCAV'09, LSAP'10, PSTI'10, AVOCS'10, 3PGCIC'10, ADCIT (Book Chapter)
- 4 submitted articles (including 2 multi-site publications)

## Software

- SimGrid: 7 releases (including 2 major releases)
- Visualization: 2 releases of Triva
- Automatic Platform Mapping: release of MintCar and UMCTool
- Synthetic Platform Generation: release of Simulacrum

## Dissemination

- 2 Tutorials: HPCS'10, CLCAR'10
- 2 Invited talks: P2P'09, RGE
- 2 SuperComputing presence: @INRIA booth in 2009 and 2010
- 3-day Workshop: The SimGrid User Days (SUD)

# SimGrid User Days

- 3 days in April 2010 in Cargese (plus a plenary meeting the day before)
  - 39 participants (including 19 invited users)
    CERN, Univ. Antwerp, Univ. Neuchâtel, Univ. Pôrto Alegre
  - 25 talks (including 10 from users)
- Intense activity period for the project:
  - Dissemination and user feedback at days
  - Team meetings for permanents and coding parties for temporaries at nights
- To be renewed soon!

# USS SimGrid as a Flagship

- Collaboration with the ANR CIP project
  - Use SimGrid to study workload characterization based on static analysis
  - Related to WP2.3 and WP6.1; 1 common meeting
- Collaboration with the ANR SPREADS project
  - Related to WP1 on scalable models; 1 common meeting
- PHC Tournesol with the University of Antwerp (2k – 2y)
  - On Scalable routing related to WP1.1; 2 weeks visit
- PICS CNRS Hawai'i/Villeurbanne (15k – 3y)
  - On on-line and off-line MPI simulation (WP2.3 and WP6.1); 2× 1-week visit
- Institut des Grilles/Aladdin project SimGlite (5k – 1y)
  - Simulation/emulation of a gLite production grid
  - Related to WP2.3 interception methods; 1 common meeting
- Institut des Grilles/Aladdin project SimData (5k – 1y)
  - Simulation of the distributed data management infrastructure of CERN
  - Requires new scalable storage models in WP1; 1 common meeting
- INRIA ADT (80k – 2y)
  - 2 × 2-year engineer positions to improve the usability of SimGrid

# Conclusion and Open questions

## Answers to good questions lead to new questions

- The work planned in this project will be done on time
- But these developments gave us new ideas about going even further
- These new ideas cannot be tackled in the time frame and will remain open

## Scalability

- Time parallel: split timeline to parallelize further
- Fluid simulation: aggregate behaviours of groups of processes

## Validity

- Endless quest: other application domains can be (even) more challenging

## Usability

- Design of experiment: automatically determine the runs answering a question
- Open science: log experiment campaigns, share them, improve other's ones

## Example of domain of application: exascale

- Programming the future supercomputers of billions of cores
- Combine the need of extreme scalability with meticulous validity