Simulation of Very Large Scale Computing Systems (Toward Exascale and Clouds...)

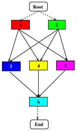
Arnaud Legrand
CNRS/INRIA MESCAL/University of Grenoble

The SimGrid Team

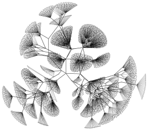
November 20, 2012

Large-Scale Distributed Systems Science?

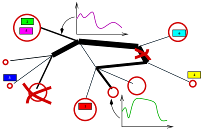
Idea to test



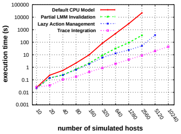
System Model



Experimental setup

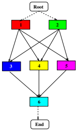


Scientific results



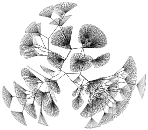
Large-Scale Distributed Systems Science?

Idea to test



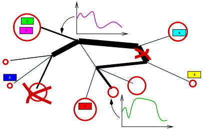
+

System Model



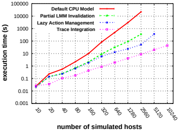
+

Experimental setup



⇒

Scientific results

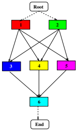


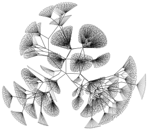
Requirements for a Scientific Approach

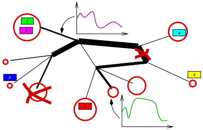
- Reproducible results
  - You can read a paper, reproduce a subset of its results and improve
- Standard tools and methodologies
  - Grad students can learn their use and become operational quickly
  - Experimental scenario can be compared accurately

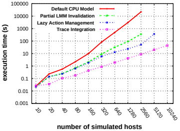
Large-Scale Distributed Systems Science?

Idea to test + System Model + Experimental setup ⇒ Scientific results









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

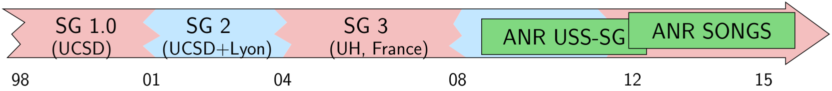
Current practice in the field: quite different

- Very little common methodologies and tools, large load of (ad-hoc) tools
  - WhateverSim and Sim-*
  - Few are really usable: Diffusion, Software Quality Assurance, Long-term availability
  - Most rely on straightforward models with no validity assessment
- Experimental settings rarely detailed in literature

The SimGrid Project

Project Purpose

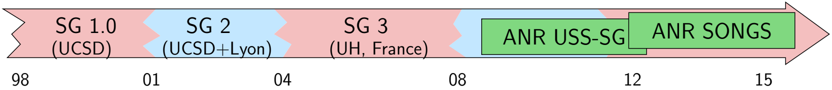
- Allow a scientific approach of Large-Scale Distributed Systems simulation
- Propose ready to use tools encouraging methodological best practices
- Methodological outcomes not specific to simulation



The SimGrid Project

Project Purpose

- Allow a scientific approach of Large-Scale Distributed Systems simulation
- Propose ready to use tools encouraging methodological best practices
- Methodological outcomes not specific to simulation



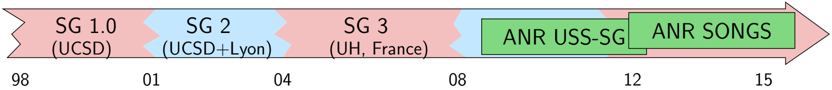
Main Challenges (a.k.a Outline of the talk)

- Validity: Get realistic results (controlled experimental bias)
- Scalability: Simulate *big enough* problems *fast enough*
- Associated tools: campaign mgmt, result analysis, settings generation, ...
- Applicability: If it doesn't simulate what is important to you, it's void

The SimGrid Project

Project Purpose

- Allow a scientific approach of Large-Scale Distributed Systems simulation
- Propose ready to use tools encouraging methodological best practices
- Methodological outcomes not specific to simulation



Main Challenges (a.k.a Outline of the talk)

- Validity: Get realistic results (controlled experimental bias)
- Scalability: Simulate *big enough* problems *fast enough*
- Associated tools: campaign mgmt, result analysis, settings generation, ...
- Applicability: If it doesn't simulate what is important to you, it's void

The SimGrid Framework as a Scientific Instrument

- Open source, Validated, Scalable, Usable, Modular, Portable
- Grounded +100 papers; 100 members on simgrid-user@; Open Source
- Simulates real programs (using specific API), not models; C, Java, Lua, Ruby

Outline

- Validity
- Scalability
- Associated Tools
- Future Directions

Validity Challenge

SotA: 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 wrong on heterogeneous platforms
- *Validity evaluation:* tricky, requires meticulous attention & sound methodology

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

Network Communication Models

Packet-level simulation

Networking community has standards, many popular open-source projects (NS, GTneTS, OmNet++,...)

- full simulation of the whole protocol stack
- complex models $\rightsquigarrow$ hard to instantiate
- inherently **slow**
- beware of simplistic packet-level simulation

Along the same lines: Weaver and MsKee, *Are Cycle Accurate Simulations a Waste of Time?*, Proc. of the Workshop on Duplicating, Deconstruction and Debunking, 2008

Network Communication Models

Packet-level simulation

Networking community has standards, many popular open-source projects (NS, GTneTS, OmNet++,. . .)

- full simulation of the whole protocol stack
- complex models $\leadsto$ hard to instantiate
- inherently **slow**
- beware of simplistic packet-level simulation

Along the same lines: Weaver and MsKee, *Are Cycle Accurate Simulations a Waste of Time?*, Proc. of the Workshop on Duplicating, Deconstruction and Debunking, 2008

Delay-based models

The simplest ones. . .

- communication time = constant delay, statistical distribution, LogP
  $\leadsto$($\Theta(1)$ footprint and $O(1)$ computation)
- coordinate based systems to account for geographic proximity
  $\leadsto$($\Theta(N)$ footprint and $O(1)$ computation)

Although very scalable, these models ignore network congestion and typically assume large bissection bandwidth

Network Communication Models (cont'd)

Flow-level models A communication (flow) is simulated as a single entity

$$T_{i,j}(S) = L_{i,j} + S/B_{i,j}, \text{ where } \begin{cases} S & \text{message size} \\ L_{i,j} & \text{latency between } i \text{ and } j \\ B_{i,j} & \text{bandwidth between } i \text{ and } j \end{cases}$$

Estimating $B_{i,j}$ requires to account for interactions with other flows

Network Communication Models (cont'd)

Flow-level models A communication (flow) is simulated as a single entity

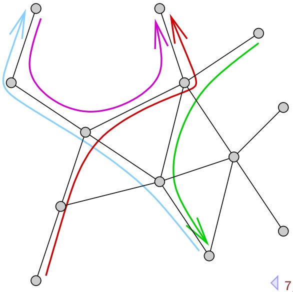
$$T_{i,j}(S) = L_{i,j} + S/B_{i,j}, \text{ where } \begin{cases} S & \text{message size} \\ L_{i,j} & \text{latency between } i \text{ and } j \\ B_{i,j} & \text{bandwidth between } i \text{ and } j \end{cases}$$

Estimating $B_{i,j}$ requires to account for interactions with other flows

Assume steady-state and **share bandwidth** every time a new flow appears or disappears

Setting a set of flows $\mathcal{F}$ and a set of links $\mathcal{L}$

Constraints For all link j: $\displaystyle\sum_{\textbf{if flow i uses link j}} \rho_i \leq C_j$



Network Communication Models (cont'd)

Flow-level models A communication (flow) is simulated as a single entity

$$T_{i,j}(S) = L_{i,j} + S/B_{i,j}, \text{ where } \begin{cases} S & \text{message size} \\ L_{i,j} & \text{latency between } i \text{ and } j \\ B_{i,j} & \text{bandwidth between } i \text{ and } j \end{cases}$$

Estimating $B_{i,j}$ requires to account for interactions with other flows

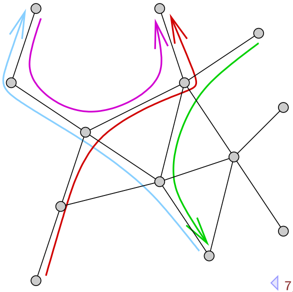
Assume steady-state and **share bandwidth** every time a new flow appears or disappears

Setting a set of flows $\mathcal{F}$ and a set of links $\mathcal{L}$

Constraints For all link j: $\sum_{\text{if flow i uses link j}} \rho_i \leq C_j$

Objective function

- Max-Min $\max(\min(\rho_i))$
- or other fancy objectives e.g., Reno $\sim \max(\sum \log(\rho_i))$



SimGrid Validity Results

SimGrid validity: Research focus since 2002

- 2002 *Sound model* proposed ⇒ Validity *checked on a few simple scenarios*.
- 2007- *Error evaluation* starts ⇒ Identify (and solve) model's weaknesses

SimGrid Validity Results

SimGrid validity: Research focus since 2002

- 2002 *Sound model* proposed $\Rightarrow$ Validity *checked on a few simple scenarios*.
- 2007- *Error evaluation* starts $\Rightarrow$ Identify (and solve) model's weaknesses

Settings: *Synthetic* App. + *Synthetic* WAN. Compare against *GTNetS*

- Errors were hunted down + unexpected phenomenon were understood
- Sharing mechanism from theoretical literature experimentally proved wrong

$\rightsquigarrow$ The model and its instanciation were considerably improved

- SimGrid and packet-level simulators now mostly diverge in extreme cases

SimGrid Validity Results

SimGrid validity: Research focus since 2002

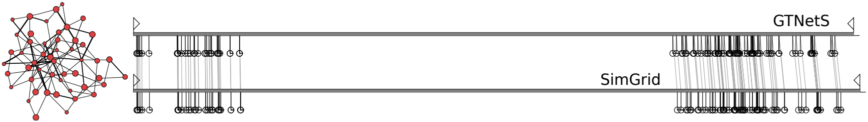
- 2002 *Sound model* proposed ⇒ Validity *checked on a few simple scenarios*.
- 2007- *Error evaluation* starts ⇒ Identify (and solve) model's weaknesses

Settings: *Synthetic* App. + *Synthetic* WAN. Compare against *GTNetS*

- Errors were hunted down + unexpected phenomenon were understood
- Sharing mechanism from theoretical literature experimentally proved wrong

⇝ The model and its instanciation were considerably improved

- SimGrid and packet-level simulators now mostly diverge in extreme cases



SimGrid Validity Results

SimGrid validity: Research focus since 2002

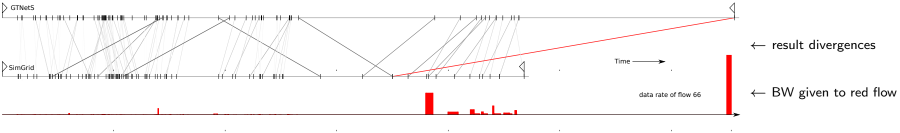
- 2002 *Sound model* proposed ⇒ Validity *checked on a few simple scenarios*.
- 2007- *Error evaluation* starts ⇒ Identify (and solve) model's weaknesses

Settings: *Synthetic* App. + *Synthetic* WAN. Compare against *GTNetS*

- Errors were hunted down + unexpected phenomenon were understood
- Sharing mechanism from theoretical literature experimentally proved wrong

⇝ The model and its instanciation were considerably improved

- SimGrid and packet-level simulators now mostly diverge in extreme cases



In this scenario, GTNetS and SG agree on termination date of most flows. The most diverging gets no bandwidth for a while although all others are done.

SimGrid Validity Results

SimGrid validity: Research focus since 2002

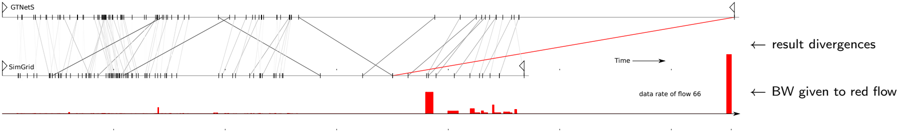
- 2002 *Sound model* proposed ⇒ Validity *checked on a few simple scenarios*.
- 2007- *Error evaluation* starts ⇒ Identify (and solve) model's weaknesses

Settings: *Synthetic* App. + *Synthetic* WAN. Compare against *GTNetS*

- Errors were hunted down + unexpected phenomenon were understood
- Sharing mechanism from theoretical literature experimentally proved wrong

⇝ The model and its instanciation were considerably improved

- SimGrid and packet-level simulators now mostly diverge in extreme cases



In this scenario, GTNetS and SG agree on termination date of most flows. The most diverging gets no bandwidth for a while although all others are done.

Such **fluid models can account** for TCP key characteristics

- slow-start
- flow-control limitation
- RTT-unfairness
- cross traffic interference

They are a very reasonable approximation for most LSDC systems

What about HPC?

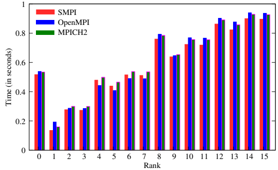
Going Further: Real App. +LAN $\leadsto$ SMPI

- Good prediction for short messages is crucial
  $\leadsto$ mixture of piecewise linear, "LogP", and flow-based modeling
- Accurately modeling MPI semantic (asynchronous & collectives ops) is tricky
  $\leadsto$ looked at the code of openMPI and reproduce
- Need to account for MPI overhead; what is Real with several MPI implems?
  $\leadsto$ account for such overheads through benchmarking

What about HPC?

Going Further: Real App. +LAN $\leadsto$ SMPI

- Good prediction for short messages is crucial
  $\leadsto$ mixture of piecewise linear, "LogP", and flow-based modeling
- Accurately modeling MPI semantic (asynchronous & collectives ops) is tricky
  $\leadsto$ looked at the code of openMPI and reproduce
- Need to account for MPI overhead; what is Real with several MPI implems?
  $\leadsto$ account for such overheads through benchmarking



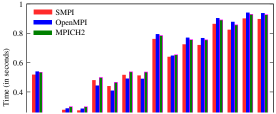
Taking resource sharing into account

- Rather good (visual) accuracy
- Our "error" $\approx$ difference between runtimes
- This is only one collective

What about HPC?

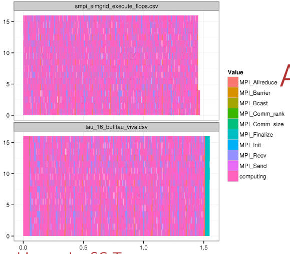
Going Further: Real App. +LAN $\rightsquigarrow$ SMPI

- Good prediction for short messages is crucial
  $\rightsquigarrow$ mixture of piecewise linear, “LogP”, and flow-based modeling
- Accurately modeling MPI semantic (asynchronous & collectives ops) is tricky
  $\rightsquigarrow$ looked at the code of openMPI and reproduce
- Need to account for MPI overhead; what is Real with several MPI implems?
  $\rightsquigarrow$ account for such overheads through benchmarking



Taking resource sharing into account

- Rather good (visual) accuracy
- Our “error” $\approx$ difference between runtimes
- This is only one collective



A “difficult” workload: Sweep3D

- Lots of ridiculously small messages and computations
- We do not only compare total time but also state distribution
- More complex apps are on their way

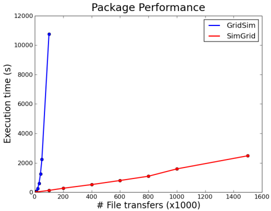
Outline

- Validity
- Scalability
- Associated Tools
- Future Directions

Scalability Challenge

Situation in 2009

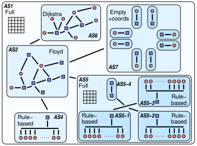
- Timings from CERN guys

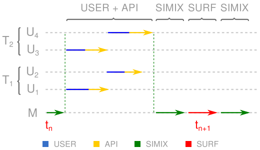


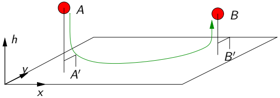
- Maximal amount of user processes
  - GridSim: 10,922 (hard limit)
  - SimGrid: 200k (memory limit, 4Gb)
- But needs of the users:
  - CERN: 300 × bigger than that (10 days/run)
  - BOINC: 600k volatile hosts over a year
- PeerSim simulates millions of processes
  - but with simplistic models only

Scalability Improvements

- Compact Routing Representation and Lazy Evaluation
- Parallel and distributed simulation
- Simpler models: Coordinate-based and Last-mile models



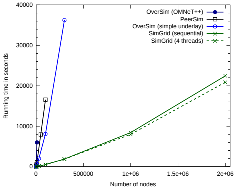




$d_{A,B} = \|A'B'\| + h_A + h_B$

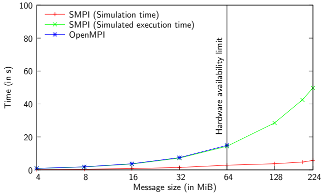
SimGrid Scalability Results

Millions of small processes (P2P)



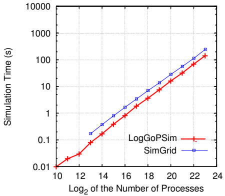
Chord simulation

Dozen of huge processes



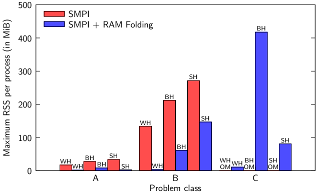
Binomial scatter with 16 processes

HPC "Workload"



MPI Broadcast: SG vs. LogGopSim
SG uses a hierarchical platform

Hundreds of large processes



DT with up to 448 processes

Outline

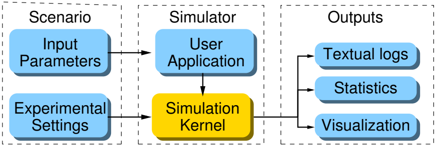
- Validity
- Scalability
- Associated Tools
- Future Directions

Associated Tools Challenge

Workflow to any Simulation Experiment

1. Prepare the scenarios
2. Launch thousands of runs
3. Post-process and analyze results

⇝ Each simulation is only a brick



- SotA: Most frameworks come with *ad hoc* tools (many are *demowares*)
  - Build a *demoware* is easy, ease understanding is harder

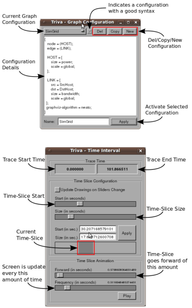
The SimGrid Ecosystem

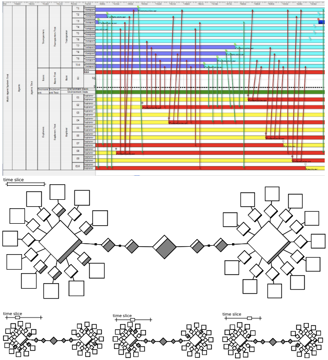
1. Workload generation:
   - Platforms: Simulacrum (generation), PDA (archive) and MintCAR (mapping)
   - Applicative Workload: Tau-based trace collection + replay
   - Background Workload: Pilgrim (trace aggregation tool)
2. Campaign management: Workflow engine
3. Single simulation analysis: Visualization
   - Builds upon separate established projects: Triva and Paje

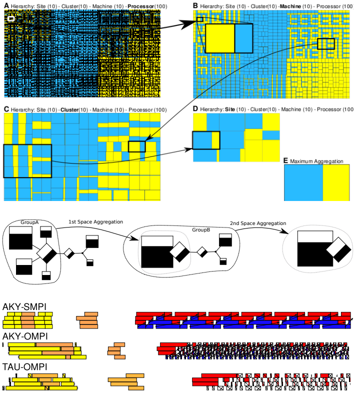
SimGrid Visualization Results

- Generic Visualization: map between trace variable to graphic objects;
- Right Representation: gantt charts, tree-maps, graphs
- Scalable tools: avoid visualization artifacts with sound aggregation
- Easy navigation in space and time: selection, aggregation, animation
- Trace comparison: Early work on trace diffing

Lucas Schnorr: Paje, Triva, **Viva**







Outline

- Validity
- Scalability
- Associated Tools
- Future Directions

And now?

1998-2001 Baby steps: Factorize some code between PhD students in scheduling

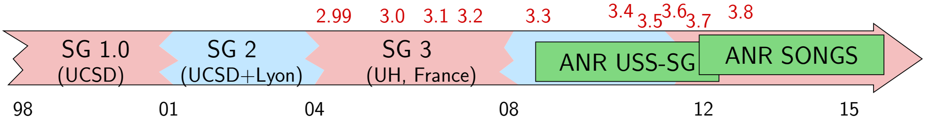
2001-2003 Infancy: CSP and improved models

2003-2008 Teenage: Performance, validity, multi-APIs

2008-2011 Maturation: Scope increase to P2P, visualization

2012- Spreading:

- Scope extension to HPC and Cloud
- Improve asociated tools: visualization, campaign management
- ANR INFRA SONGS project



SONGS (Simulation Of Next Generation Systems)

Lessons learned from past projects

- Much emphasis on methodology, but users' concerns are important too
- Science pulled by needs, not pushed by abilities: Use-Case Driven Research

Main Goal: Making SimGrid usable in 2 more domains

- Task 1: [Data]Grid (with CERN)
- Task 2: Peer-to-Peer and Volunteer Computing (with BOINC)
- Task 3: IaaS Clouds
- Task 4: High Performance Computing

Factorize developments on simulation pillars

- Task 5: Simulation Kernel (efficient and standard simulations)
- Task 6: Concepts and Models (power, storage, CPU/Mem, networks, volatility)
- Task 7: Analysis and Visualization
- Task 8: Experimental Methodology (DoE, Open science, Campaign magmt)

SONGS: Simulating IaaS Clouds

Envisioned Provider-Side Studies

- Anticipated provisioning of VMs to face peak demands
- Allocation algorithms to map VMs to physical hosts
- Placement of VM images to reduce VM startup and migration
- Metrics: Performance, Energy, Resource usage, Economics
- API: VM operations (start/stop/migrate), images storage, SLA

Envisioned Client-Side Studies

- Leverage cloud billing model to get the best performance at the best prices
- Evaluate trade-off between price and performance according to the workload
- Force provider to SLA violations to get free resources ;)
- API: EC2 (de facto standard)
- Missing models:
  APIs elements, non-CPU-bound tasks, resource sharing between VMs

SONGS: Simulating HPC systems

Challenge: Simulate complex apps running on modern HPC platforms

- Huge modeling task, daunting validation challenge

Missing Models and Concepts

- CPU model: Flops count $\rightsquigarrow$ multicores w/ complex mem. hierarchies, GPU
- Network: Ethernet only $\rightsquigarrow$ infiniband at least
- Memory resource to be added (cache effects, NUMA archs)
- Energy need DVFS API and flexible composition, I/Os...

Envisioned Studies

- Classical MPI applications
- Challenging MPI apps (highly optimized wrt memory and CPU), StarPU
- ExaScale: Capacity planning for 100,000+ ARM processors (easier than Intel)

Risks and backups

- Previous experience with SMPI (simulated MPI)
- Huge task split in several steps; doing *some* steps would be something

Conclusion

Main concerns of the SimGrid project

- Validity: Get realistic results (controlled experimental bias)
- Scalability: Simulate *fast enough* problems *big enough*
- Associated tools: campaign mgmt, result analysis, settings generation, ...
- Applicability: If it doesn't simulate what is important to you, it's void

Conclusion

Main concerns of the SimGrid project

- Validity: Get realistic results (controlled experimental bias)
- Scalability: Simulate *fast enough* problems *big enough*
- Associated tools: campaign mgmt, result analysis, settings generation, . . .
- Applicability: If it doesn't simulate what is important to you, it's void

Potential collaborations in the joint lab

LogGOPSim Loading GOAL in SimGrid is trivial; Would allow to simulate seamlessly network hierarchy and contention

- Is the GOAL formalism still relevant? LogGS linear regressions?
- What about injecting system noise? Failures?
- Using SG to evaluate topology-aware collective communications?

Conclusion

Main concerns of the SimGrid project

- Validity: Get realistic results (controlled experimental bias)
- Scalability: Simulate *fast enough* problems *big enough*
- Associated tools: campaign mgmt, result analysis, settings generation, . . .
- Applicability: If it doesn't simulate what is important to you, it's void

Potential collaborations in the joint lab

LogGOPSim Loading GOAL in SimGrid is trivial; Would allow to simulate seamlessly network hierarchy and contention

- Is the GOAL formalism still relevant? LogGS linear regressions?
- What about injecting system noise? Failures?
- Using SG to evaluate topology-aware collective communications?

BigSim Seems more resource demanding but I haven't tried it yet

- May want to benefit from hierarchical network models?
- Has nice tricks for variable privatization
- Handles distributed execution while it is WIP in SG

Conclusion

Main concerns of the SimGrid project

- Validity: Get realistic results (controlled experimental bias)
- Scalability: Simulate *fast enough* problems *big enough*
- Associated tools: campaign mgmt, result analysis, settings generation, ...
- Applicability: If it doesn't simulate what is important to you, it's void

Potential collaborations in the joint lab

LogGOPSim Loading GOAL in SimGrid is trivial; Would allow to simulate seamlessly network hierarchy and contention

- Is the GOAL formalism still relevant? LogGS linear regressions?
- What about injecting system noise? Failures?
- Using SG to evaluate topology-aware collective communications?

BigSim Seems more resource demanding but I haven't tried it yet

- May want to benefit from hierarchical network models?
- Has nice tricks for variable privatization
- Handles distributed execution while it is WIP in SG

Visualization tools (UFRGS, Brazil) need **new** tools with **aggregation** capabilities