Simulation Of Next Generation Systems

SONGS

Bordeaux (Cepage): L. Eyraud, O. Beaumont, N. Bonichon, F. Mathieu, (Runtime): D. Barthou, B. Goglin, A. Guermouche

Grenoble (Mescal): A. Legrand, D. Kondo, J.-F. Méhaut, J.-M. Vincent

Nancy (Algorille): M. Quinson, L. Nussbaum

Nantes (Ascola): A. Lèbre

Nice (Mascotte): O. Dalle, H. Renard

Villeurbanne (CCIN2P3): F. Suter, P.-E. Brinette, F. Desprez

Strasbourg (ICPS): S. Genaud, J. Gossa

ANR (INFRA 2011)

18 janvier 2012, Paris

Our Scientific Objects: Distributed Systems

Scientific Computing: High Performance Computing / Computational Grids

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

Cloud Computing

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

P2P Systems

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

Systems already in use, but characteristics hard to assess

- Performance: makespan, economics, energy, ... $\leftarrow$ context of this project
- Correction: absence of crash, race conditions, deadlocks and other defects

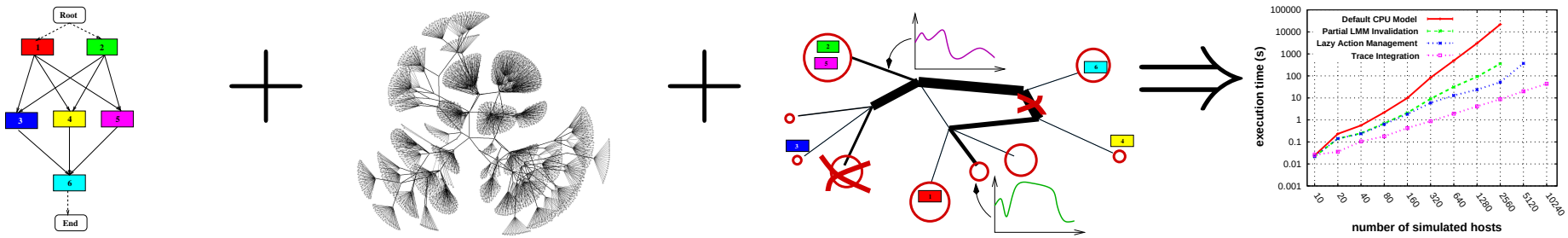
Assessing Distributed Applications' Performance

Most Performance Studies are conducted through Experimentation

- Experimental Facilities: real applications on real platforms *(in vivo)*
- Emulation: real applications on models of platforms *(in vitro)*
- Simulation: models (prototypes) of applications on system's models *(in silico)*

Simulating Distributed Systems ← context of this project

Idea to test | System Model | Experimental setup | Scientific results



Simulation's Advantages

- Less simplistic than proposed theoretical models (which are useful too)
- Better XP control ($\rightsquigarrow$ 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

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

State of the Art of Distributed Apps Simulation

Each (sub-)domain came up with Numerous Simulators

	Dom	CPU	Disk	Network	Application	Scale
ns2	Networking	None	None	Packet-level	Coarse d.e.s.	<1,000
GTNetS	Networking	None	None	Packet-Level	Coarse d.e.s.	<177,000
OptorSim	(Data)Grid	Analytic	Amount. No perf. info	Analytic. buggy	Coarse d.e.s.	Few 1000
GridSim	Grid	Analytic	Analytic	Wormhole	Coarse d.e.s.	Few 1000
PlanetSim	P2P	None	None	Constant time	Coarse d.e.s.	100,000
PeerSim	P2P	None	None	None	State machine	1,000,000
SimGrid	Grid, VC, P2P, HPC, cloud, ...	Analytic	None	Analytic or Packet-level (GTNetS, NS3)	Coarse d.e.s. or emulation	Few 1,000,000

Simulation for distributed applications still in infancy

- Compared for example to Hardware Design or Networking Communities
- Lot of home-made tools; No standard methodology
- Few publish their tools for others to use
- Very few support other people's use: genericity, stability, portability, doc, ...
- (Almost) nobody validate their tools

The SimGrid Project

Project Purpose

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

Main Challenges

- 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

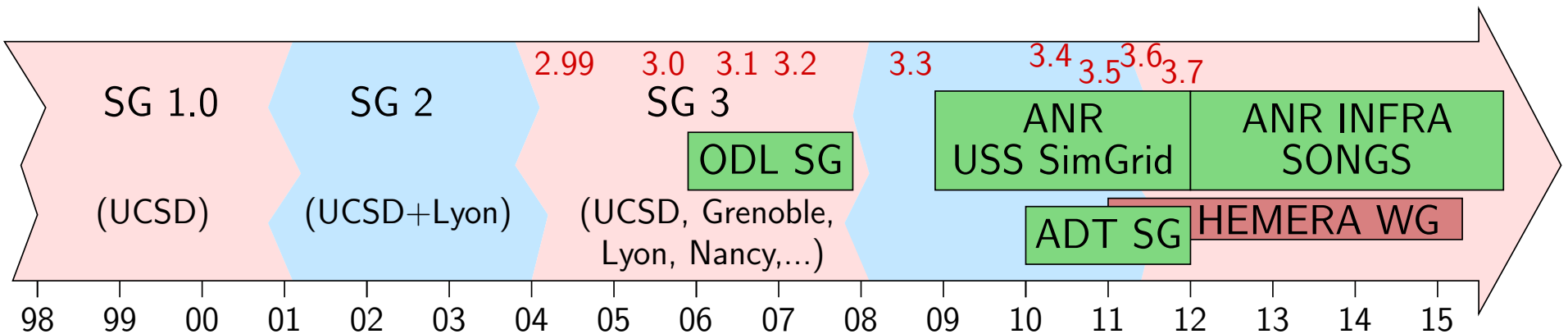
The SimGrid Framework as a Scientific Instrument

- 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
- Contains prototypal model-checker (but off topic for the SONGS project)

SimGrid History

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

- Features: Platform heterogeneity and dynamicity (traces)
- Done by H. Casanova; Scope: limited to scheduling at UCSD

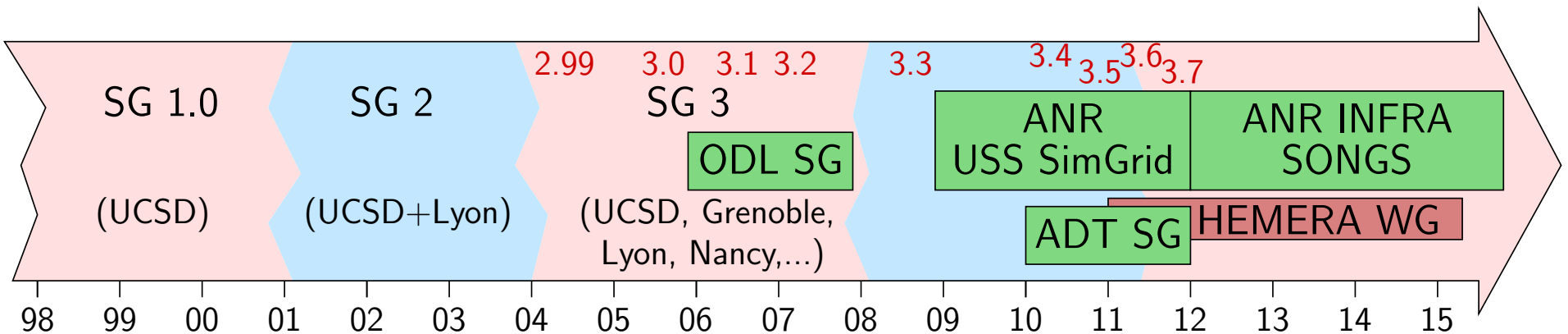


SimGrid History

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

2001-2003 Infancy:

- Allow distributed scheduling: A. Legrand added CSPs on top of existing
- Model validity questionable: L. Marchal implemented analytical models
- Scope: UCSD + ENS-Lyon and few others



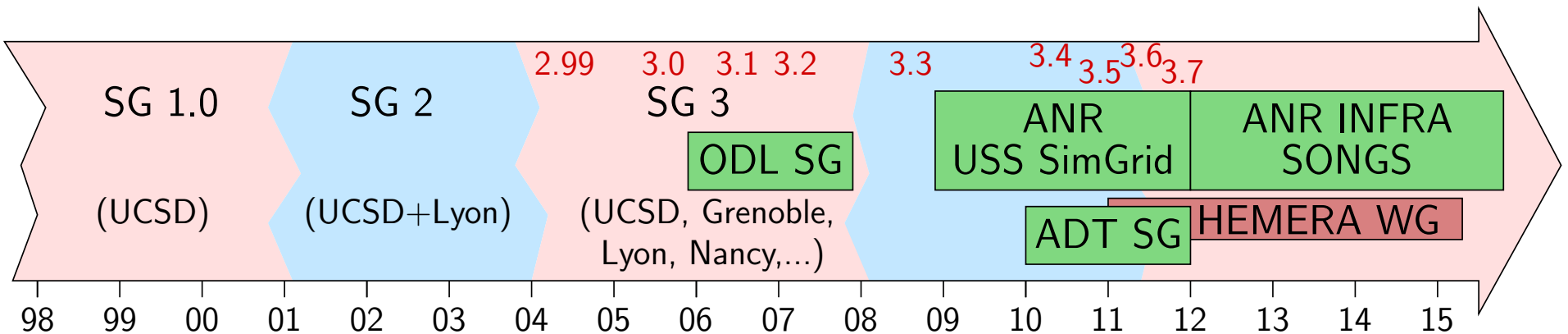
SimGrid History

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

2001-2003 Infancy: CSP and improved models

2003-2008 Teenage:

- Performance improvement: rewrite kernel from scratch (SURF)
- Validity quest: Hunt (and fix) worst case scenarios
- Other interfaces: GRAS, SimDag, SMPI



SimGrid History

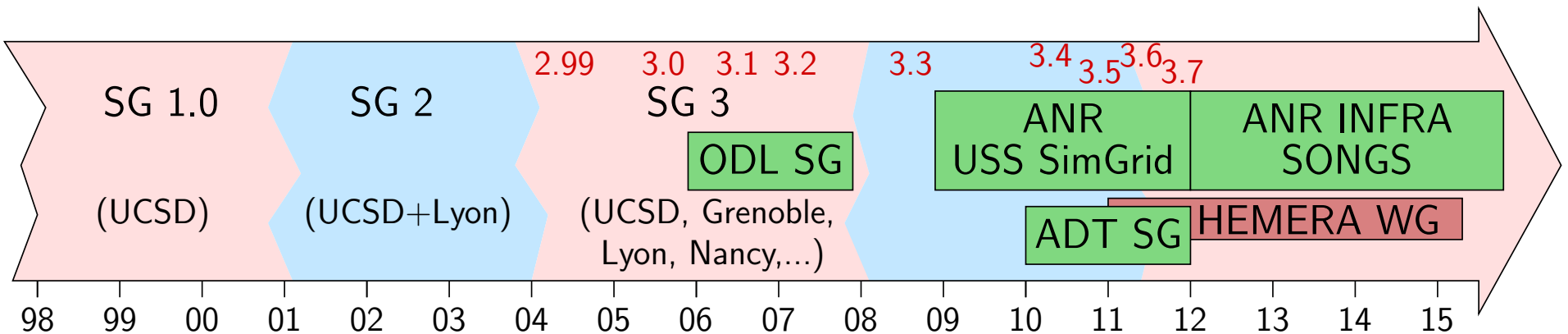
1998-2001 Baby steps: Factorize some code between PhD students in scheduling

2001-2003 Infancy: CSP and improved models

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

2008-2011 Maturation:

- Scope extension from Grid to P2P and VC: scalable to death, peer models
- Associated tools: visualization, campaign management
- ANR USS SimGrid + support from INRIA (ADT)



SimGrid History

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

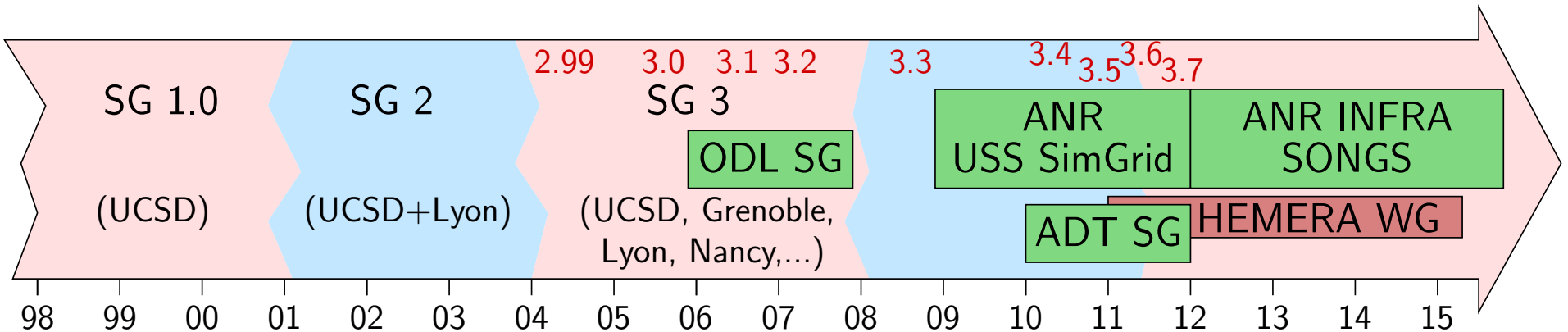
2001-2003 Infancy: CSP and improved models

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

2008-2011 Maturation: Scope increase to P2P, visualization

2012-: Taking over the world ;-)

- Scope extension to HPC, Cloud
- Associated tools: visualization, campaign management
- ANR INFRA SONGS



SimGrid Internals in a Nutshell

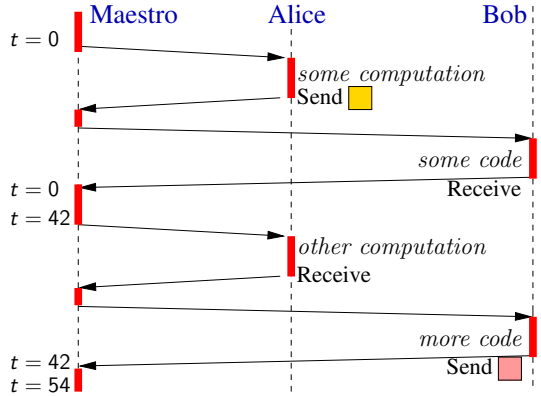
Example of user code to execute

Alice

```
(some computation)
Send "toto" to Bob
(other computation)
Receive from Bob
```

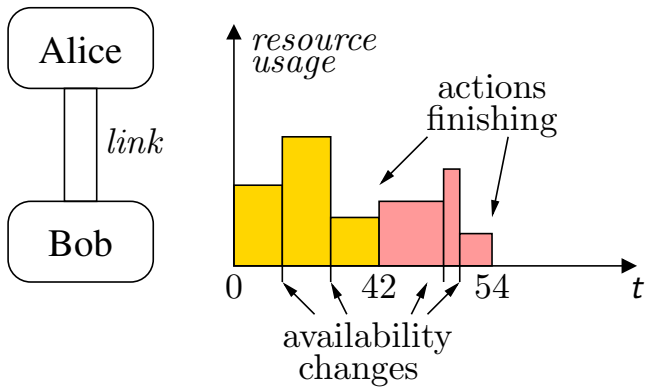
Bob

```
(some code)
Receive from Alice
(other code)
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. Update simulated clock
5. Unlock user code waiting on this action



SimGrid Usage: executive MSG example

1. Write the Code of your Agents

```
int master(int argc, char **argv) {
for (i = 0; i < number_of_tasks; i++) {
 t=MSG_task_create(name,comp_size,comm_size,data);
 sprintf(mailbox,"worker-%d",i % workers_count);
 MSG_task_send(t, mailbox);}
```

```
int worker(int ,char**){
sprintf(my_mailbox,"worker-%d",my_id);
while(1) {
  MSG_task_receive(&task, my_mailbox);
  MSG_task_execute(task);
  MSG_task_destroy(task);}
```

2. Describe your Experiment

XML Platform File

```
<?xml version='1.0'?>
<!DOCTYPE platform SYSTEM "surfxml.dtd">
<platform version="2">
<host name="host1" power="1E8"/>
<host name="host2" power="1E8"/>
...
<link name="link1" bandwidth="1E6"
                   latency="1E-2"  />
...
<route src="host1" dst="host2">
  <link:ctn id="link1"/>
</route>
</platform>
```

XML Deployment File

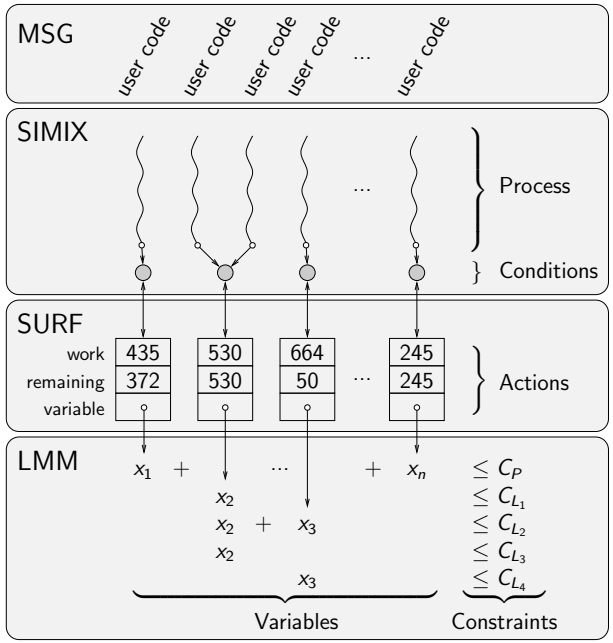
```
<?xml version='1.0'?>
<!DOCTYPE platform SYSTEM "surfxml.dtd">
<platform version="2">
<!-- The master process -->
<process host="host1" function="master">
 <argument value="10"/><!--argv[1]:#tasks-->
 <argument value="1"/><!--argv[2]:#workers-->
</process>

<!-- The workers -->
<process host="host2" function="worker">
  <argument value="0"/></process>
</platform>
```

3. Write a main gluing things together, link and run

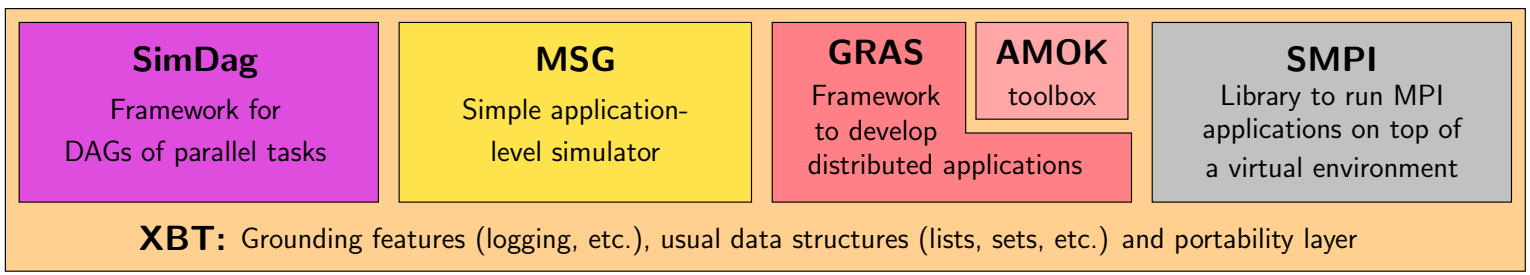
SimGrid Modules

SimGrid Functional Organization

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



SimGrid user APIs



- SimDag: heuristics as DAG of (parallel) tasks
- MSG: heuristics as Concurrent Sequential Processes (Java/Ruby/Lua bindings)
- GRAS: develop real applications, studied and debugged in simulator
- SMPI: simulate MPI codes

USS-SimGrid

ANR ARPEGE project 2008-2011 (ANR 08 SEGI 022)

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

Members

Bordeaux	(Cepage) O. Beaumont, N. Bonichon, L. Eyraud-Dubois
Lyon	(Graal/CC IN2P3) F. Desprez, M. Imbert, F. Vivien / F. Suter
Grenoble	(Mescal) A. Legrand, J.M. Vincent
Nancy	(Algorille) S. Genaud, M. Quinson
Reims	(SysCom) A. Bui, O. Flauzac, C. Rabat, L.A. Steffenel
Saclay $\rightsquigarrow$ Nice	(Asap) F. Le Fessant $\rightsquigarrow$ (Mascotte) O. Dalle

Coming next in these slides

- Details on challenges addressed in USS: Validity, **Scalability**, Tools
- The vision of the SONGS project: **Applicability**

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

SimGrid validity before USS: Research focus in SimGrid since 2002

- Several models: GTNeTs; Fast *sound* model ; Ultra-fast simplistic model
- *Sound model* proposed 10 years ago after observations and results from the network literature. Validity *checked on a few simple scenarios*.
- More thorough *error evaluation* started in 2007: in percents if TCP steady state (flows $>$ 10Mb) and latency-bound (WAN). Pretty bad otherwise.

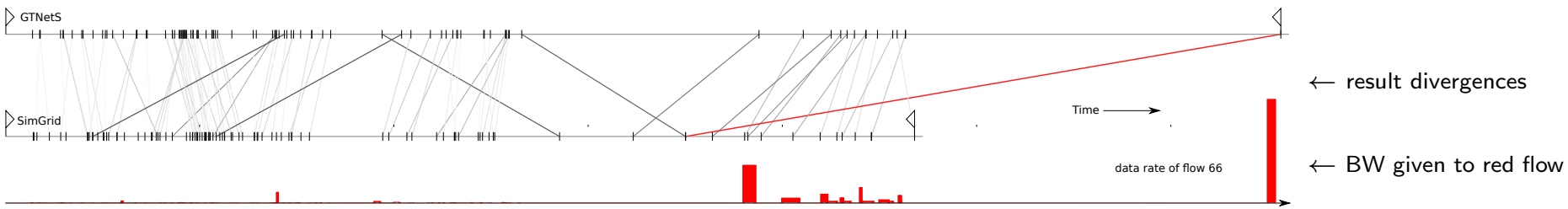
Validity: SimGrid compared to Packet-Level Tools

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
Widen validity range to flows $>$ 100Kb and WAN with small latencies

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

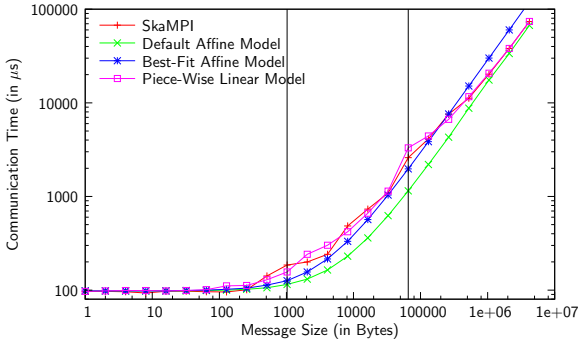


In this scenario, GTNetS and SG agree on termination date of most flows. The most diverging gets no bandwidth for a while although all others are done.

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

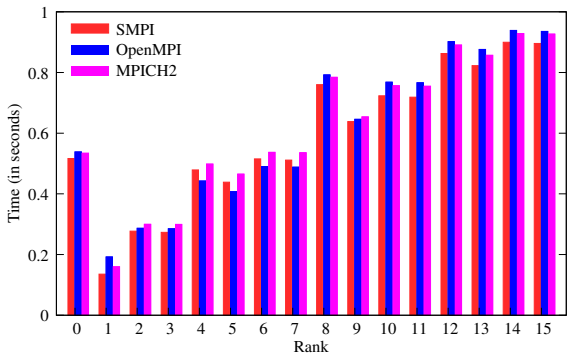
- Good prediction for short messages is crucial; Numerical instabilities deadly
- Accurately modeling MPI semantic (asynchronous & collectives ops) is tricky
- Need to account for MPI overhead; what is Real with several MPI implems?

Accuracy of MPI simulations



Timings of each communication

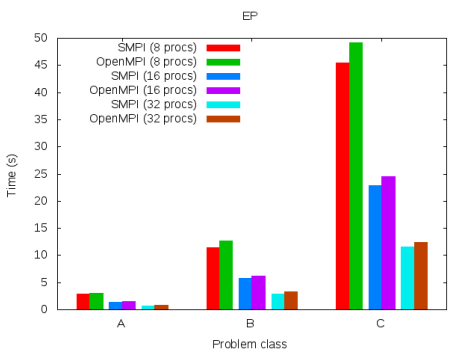
- $\lambda + size \times \tau$ not sufficient (TCP congestion)
- No affine fonction can match for all message sizes
- A 3-parts piecewise affine gives satisfying results

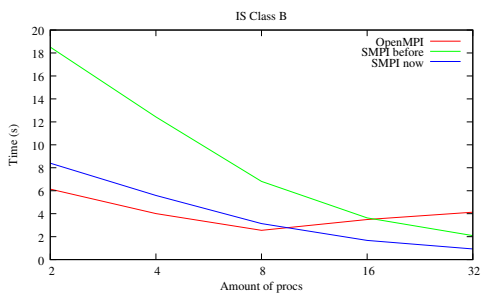


Taking resource sharing into account

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

Still a work in progress for complete MPI applications





- Performance prediction not correct
- Trashing particularly challenging
- Not perfect but still better than most other MPI simulators

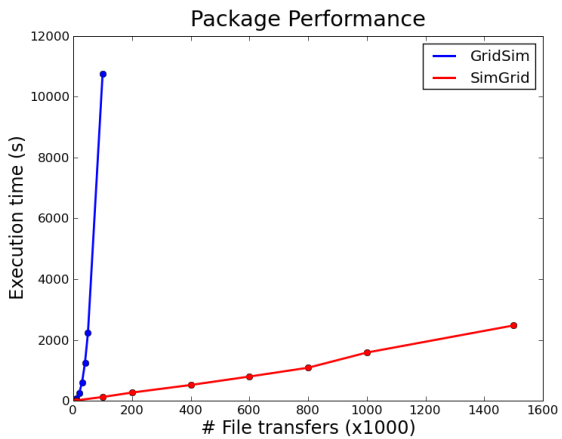
Scalability Challenge

Scalability constitutes the main objective of the USS SimGrid

- Two aspects: Big enough (large platforms) $\oplus$ Fast enough (large workload)

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

Approaches to Scalability in USS-SimGrid

Algorithmic optimization

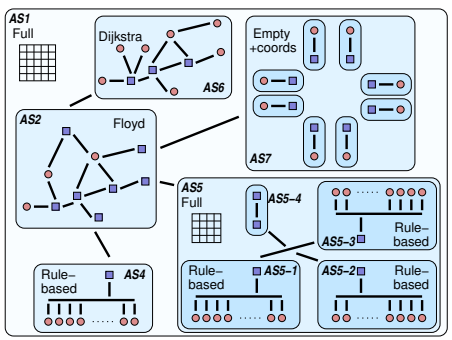
- Compact Routing Representation: From $O(n^2)$ to $O(n)$ memory consumption
- Lazy Evaluation: Arbitrary speedups on loosely coupled scenarios

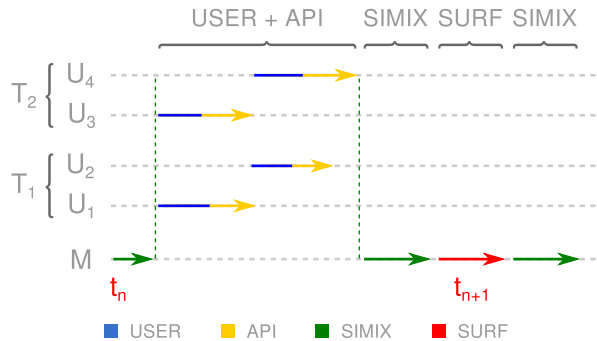
Leverage several computing units

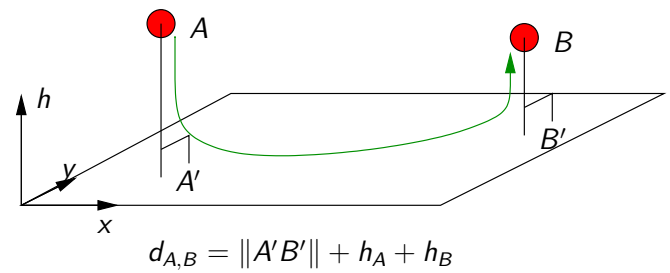
- Parallel simulation: P2P's grain so fine that classical // schema not applicable
- *Distributed simulation:* Still TBD, but not needed due to other optimizations

Simpler models (but with potential loss of realism)

- Coordinate-based: extremely efficient, but only encodes latency
- Last-mile models (Manhattan distances): very efficient; controlled realism loss



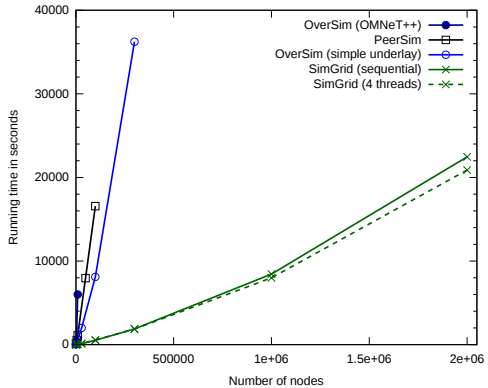




$d_{A,B} = \|A'B'\| + h_A + h_B$

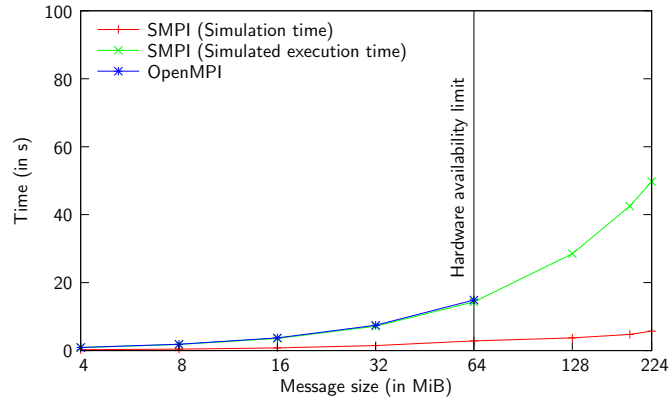
SimGrid Scalability Results

Millions of small processes



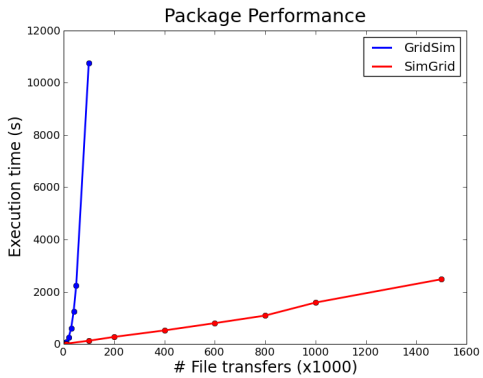
Chord simulation

Dozen of huge processes



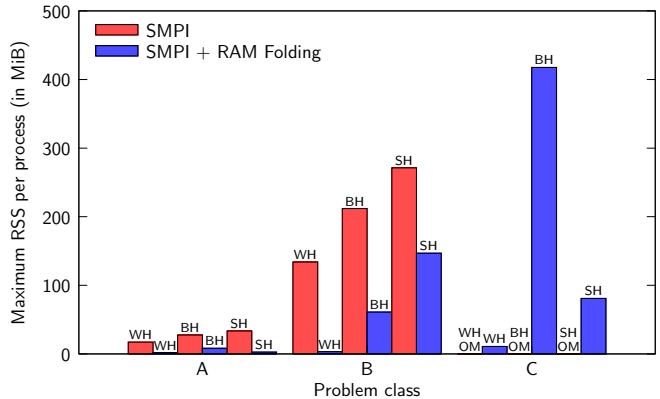
Binomial scatter with 16 processes

Large Workload



Simulation from CERN users

Hundreds of large processes



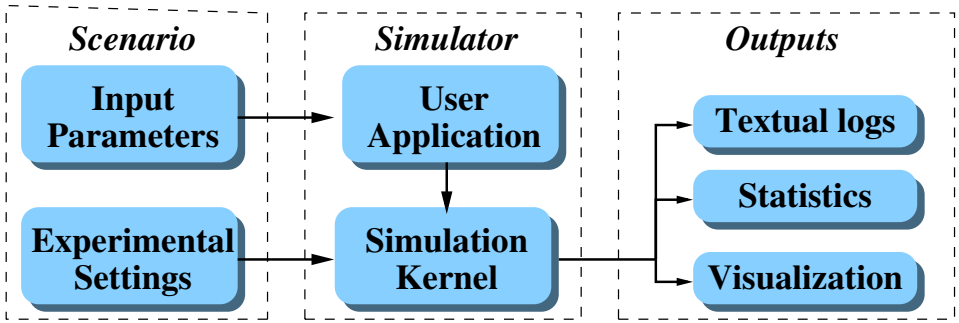
DT with up to 448 processes

Associated Tools Challenge

Workflow to any Experiments through Simulation

1. Prepare the experimental scenarios
2. Launch thousands of simulations
3. Post-processing and result analysis

$\rightsquigarrow$ Each simulation is only a brick

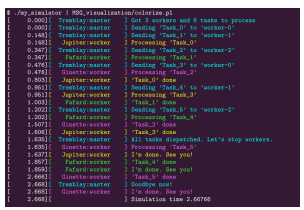


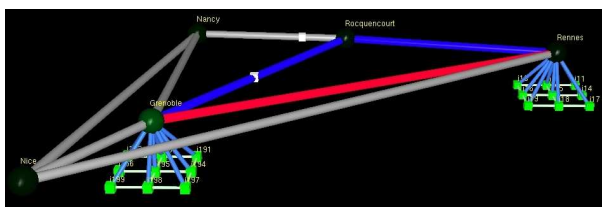
Situation before the project

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

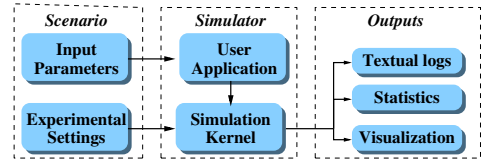
Building a *demoware* is easy. Helping understanding is harder

- Often specific to a given simulator; often scalability issues
- Show only what the authors needed (platform/app. state, tracing/profiling)



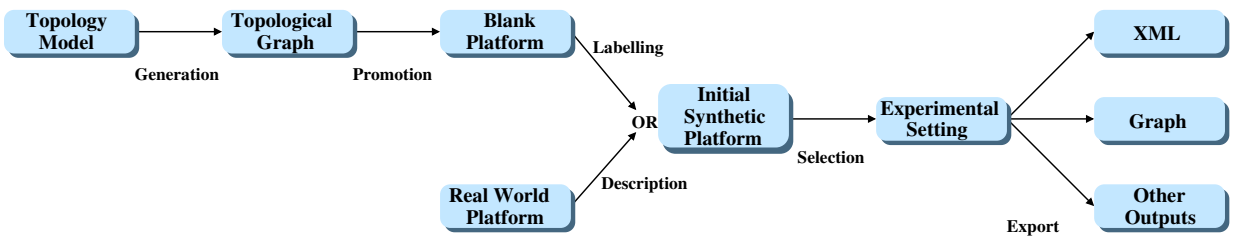


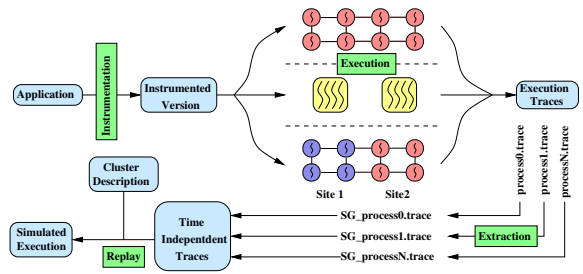
Approaches to Associated Tools in USS-SimGrid



USS-SimGrid Proposal

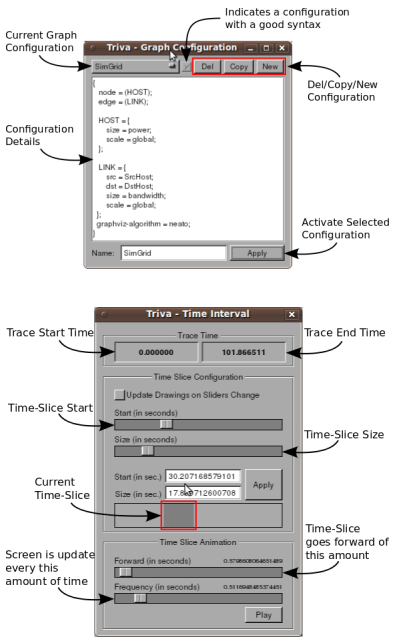
1. Workload generation:
   - Platforms: Simulacrum (generation), PDA (archive) and MintCAR (mapping)
   - Applicative Workload: Tau-based trace collection + replay
   - Background Workload: Pilgrim (trace aggregation tool)
2. Campaign management: Workflow engine
3. Single simulation analysis: Visualization
   - Builds upon separate established projects: Triva and Paje
   - Generic and dedicated to visualization: SimGrid only produces adapted traces (but SimGrid heavily modified to that extend by Triva author)

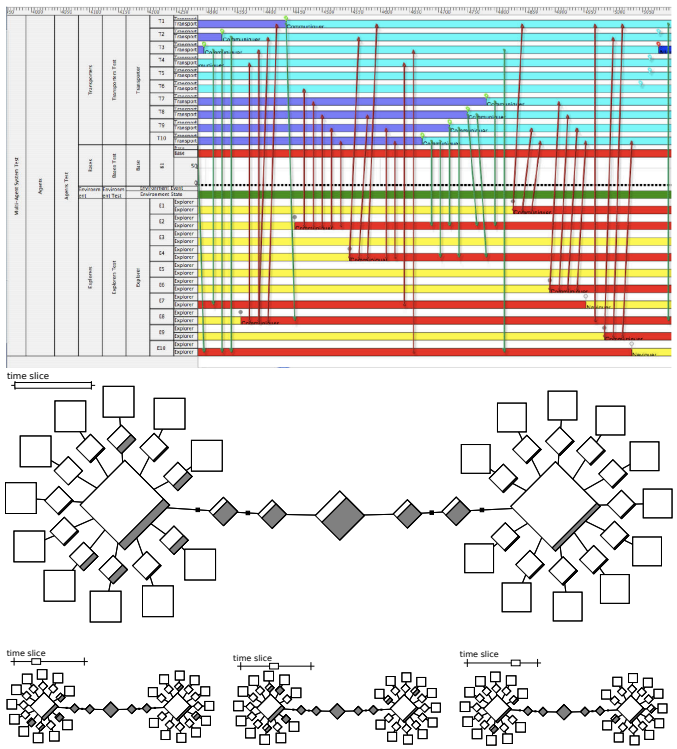


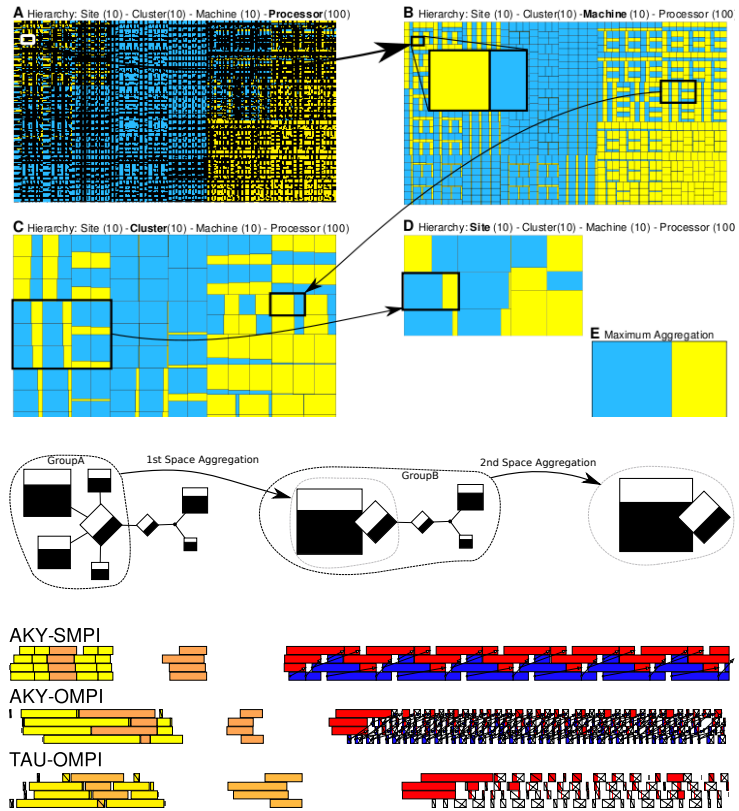


Visualizing SimGrid Simulations

- Visualization scriptable: easy but powerful configuration; Scalable tools
- Right Information: both platform and applicative visualizations
- Right Representation: gantt charts, spatial representations, tree-graphs
- Easy navigation in space and time: selection, aggregation, animation
- Easy trace comparison: Trace diffing (still partial ATM)







SONGS (Simulation Of Next Generation Systems)

Lessons drawn from USS-SimGrid

- Too much emphasis on methodology, our users' concerns are more important
- Science pulled by needs, not pushed by abilities: Use-Case Driven Research
  (you can say where you go, and when you're done; focus and motivation improved)

> *Better to have an approximate answer to the right question than a precise answer to the wrong question.* – John Tukey

Main Goal: Making SimGrid usable in 2 more domains

- Task 1: [Data]Grid (informal collab with CERN's LHC)
- Task 2: Peer-to-Peer and Volunteer Computing
- Task 3: IaaS Clouds (informal collab with IBM France and Haifa)
- Task 4: High Performance Computing

	Network		CPU	
	quantitative	qualitative	quantitative	qualitative
"Old grids"	dozen	NREN	hundreds	spare
New grids	large scale	NREN	hundreds	(multi-)clusters
P2P	large scale	Internet	large scale	spare
Clouds	large scale	WAN/LAN	thousands	(multi-)clusters
HPC	large scale	LAN	thousands	(multi-)clusters

Work on Domains

Each domain follows the same work plan

- Tx.1: Add models needed by the planned studies
  - Grids: Storage modeling
  - P2P/VC: Churn and scalable network modeling
  - IaaS Clouds: VMs, hypervisors
  - HPC: HPC networks, memory transfers
- Tx.2: Extend APIs to ease the planned studies
  - Grids: High Performance Storage System API
  - P2P/VC: Higher level API such as catalog handling
  - IaaS Clouds: Provider side, and client side
  - HPC: MPI, OpenMP, Plasma&Magma
- Tx.3: Do planned studies
  - Grids: Distributed Data mgnt for LHC; Hierarchical Storage System
  - P2P/VC: Replica Placement in VOD; Affinities in VC
  - IaaS Clouds: Study from client or provider POV; other metrics (energy)
  - HPC: exascale; memory & energy models

Coming Next

- Some more details on newly added domains (IaaS Clouds and HPC)

IaaS Clouds

Focus: Infrastructure as a Service

- No planned support for SaaS (no notion of Software in SimGrid)

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

High Performance Computing

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
- Network: Ethernet only $\rightsquigarrow$ infiniband at least
- Memory resource to be added (cache effects, NUMA archs)

Envisioned Studies

- BigDFT, SPECFEM 3D: Classical MPI applications
- MUMPS: Challenging MPI app (highly optimized wrt memory and CPU)
- ExaScale: Capacity planning for 100,000+ ARM processors (easier than AMD)

Risks and backups

- Previous experience with SMPI (simulated MPI)
- Huge task splited in several steps; doing *some* steps would be something (+easy to be more realistic than existing SotA simulators)

Work on the Simulation Pillars

- Task 5: Simulation Kernel
- Task 6: Concepts and Models
- Task 7: Analysis and Visualization
- Task 8: Support to Experimental Methodology

Work on the Simulation Pillars

- Task 5: Simulation Kernel
  - T5.1: Efficient Simulation Kernel (big enough, fast enough)
  - T5.2: Standard Simulation Kernel (DEVS)
- Task 6: Concepts and Models
  - T6.1: Power consumption (Cloud/HPC)
  - T6.2: Storage Elements (Grid/HPC/Cloud)
  - T6.3: CPU and Memory (HPC)
  - T6.4: High Performance Networks (HPC/Clouds)
  - T6.5: Volatility & Dynamicity (P2P/Clouds/HPC)
- Task 7: Analysis and Visualization
  - T7.1: Scalable Visualization
  - T7.2: Trace Comparison
- Task 8: Support to Experimental Methodology
  - T8.1: Design of experiments
  - T8.2: Campaign management
  - T8.3: Open Science

Partners (1/2)

Nancy

- M. Quinson (83%) Expert in simulation; Coordinator, **Leader WP5** $\in$WP2, 6, 7 and 8
- L. Nussbaum (25%) Expert in experimentation methodo, $\in$ WP8

Grenoble

- A. Legrand (80%) Expert in simulation; **Leader WP7**, $\in$ WP2, 5, 6, 7 and 8.
- D. Kondo (15%) Expert in workload characterization of Volunteer Computing and Clouds, $\in$ WP2, 3 and 6
- J.-F. Méhaut (25%) Expert in HPC; $\in$ WP4 and 6
- J.-M. Vincent (42%) Expert in simulation and stochastic modeling; $\in$ WP7,8

Villeurbanne

- F. Suter (50%) Expert in Grids and simulation; **Leader WP1**, $\in$ WP3, 4, 6
- P.-E. Brinette (6%) Expert in storage, $\in$ WP1
- F. Desprez (25%) Expert in Clouds and HPC, $\in$ WP3, 4 and 6

Partners (2/2)

Bordeaux

- L. Eyraud-Dubois (75%) Expert in P2P and simulation, **Leader WP6**, $\in$ WP2
- D. Barthou (25%) Expert in HPC, $\in$ WP4 and 6
- O. Beaumont (66%) Expert in P2P and optimisation algorithms, $\in$ WP2,3
- N. Bonichon (50%) Expert in P2P, **Leader WP2**
- F. Mathieu (20%) Expert in P2P, $\in$ WP2
- B. Goglin (25%) Expert in HPC, $\in$ WP4 and 6
- A. Guermouche (25%) Expert in HPC, **Leader WP4**, $\in$ WP6

Strasbourg

- S. Genaud (60%) Expert in Clouds and HPC, **Leader WP3**, $\in$ WP4
- J. Gossa (50%) Expert in Clouds, $\in$ WP3 and 7

Nantes

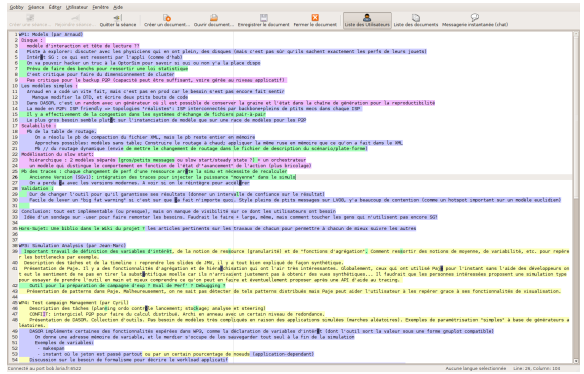
- A. Lèbre (25%) Expert in Virtualization and Storage, $\in$ WP 3 and 6

Nice

- O. Dalle (75%) Expert in simulation, **Leader WP8**, $\in$ WP5
- H. Renard (25%) Expert in scheduling, $\in$ WP8

Working Together

Meeting minutes: Gobby



- Collaborative edition of textual document
- Use wiki markup if possible

Day to day discussions: IRC and Instant Messaging

- Most of us use Instant Messaging (my contacts are on my web page)
- But point-to-point are not always enough
- We also use #simgrid and #songs on `irc.debian.org`
  send technical to #simgrid and non-sensitive scientific/politic to #songs?

Mailing lists

- Please use and abuse the mailing lists, avoid point-to-point comm by all means
- infra-songs-member: Unsorted or low traffic announcements?
- infra-songs–wp[0-8]: where the stuff happens. With cross-posting on needs

Working Together 2.0

Sharing Code and articles

- SimGrid is under git, on gforge.inria.fr
- Our article so far were in a private project on gforge, but not scalable
- A gitolite may be what we need here (volunteers to set it up?)

Keeping Track of Discussions

- We could use the wiki on `http://infra-songs.gforge.inria.fr` (but didn't really worked last time: FredS were the "main" author)
- We could use a trac system? (volunteers?)
- We could experiment with bibliography specific tools? (volunteers?)

Sharing Scientific Progresses

- First Songs Monthly Status (SMS) a while ago (yes, I intend to send a 15kb SMS each month)
- Goal: status report on each fronts: keep other informed; gather feedback
- They are written in the SVN, and collaborative edition hopped

Being visible

- Public website: `infra-songs.gforge.inria.fr` (wiki ⇒ please feed it)
- Backlinks from our pages to the sites (google bombing)

Working Together (brick and mortar)

Plenary meetings

- We should do two plenaries per year
- Goal: share pre-publication results (inform others; gather lastest feedbacks)

Sprints

- Not everybody, on several days (one week?) on a more specific point
- We have specific fundings for these, they are highly profitable (and enjoyable)
- We should do it often. We did only a (generic) sprint in USS, was fun.
- Point-to-point visits are crucial too. Please visit other sites.

SimGrid Users' Days (SUD)

- Several days, everybody + as much external users as we manage to attract
- Goal: Present news, get user feedback and orientations, convince prospective
- We have funding (and promised) to do one SUD per year in SONGS

These event types can be combined (plenary+SUD comes to mind)

Other misc administraviæ points

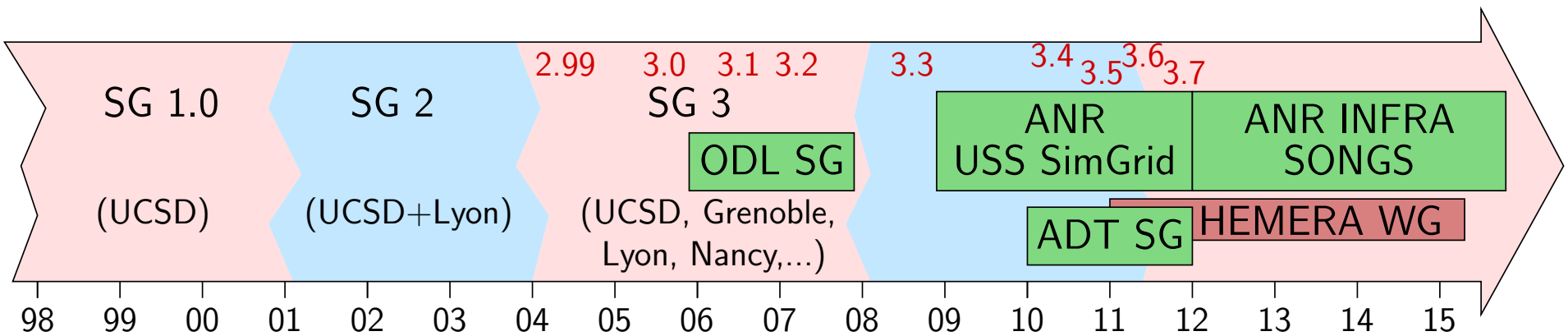
En vrac

- Signature attributive: on en est on
- Attention à recruter à temps, c'est le piège classique
- En cas de problème (ou pas), pensez à m'informer en passant par les listes
- Est ce que j'ai pensé à dire que le gros defi est de faire circuler l'information? Surtout à 30 personnes sur 8 sites...

Conclusion

Main Challenges to the Simulation of Distributed Applications

- 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



SONGS Big Picture

Main Goal: Making SimGrid usable in 2 more domains

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

Factorizing developments to the death

- 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)

And now, let the fun begin...