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

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

Coordinated by Martin Quinson (Université de Lorraine)

Lyon, January 4 2012

Our Scientific Objects: Distributed Systems

Grid Computing: Distributed infrastructure for Computational Science

- Massive systems federating numerous organizations worldwide
- Main issues: Large production infrastructures, challenging to experiment with

P2P Systems

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

Cloud Computing

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

Systems already in use, but characteristics hard to assess

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

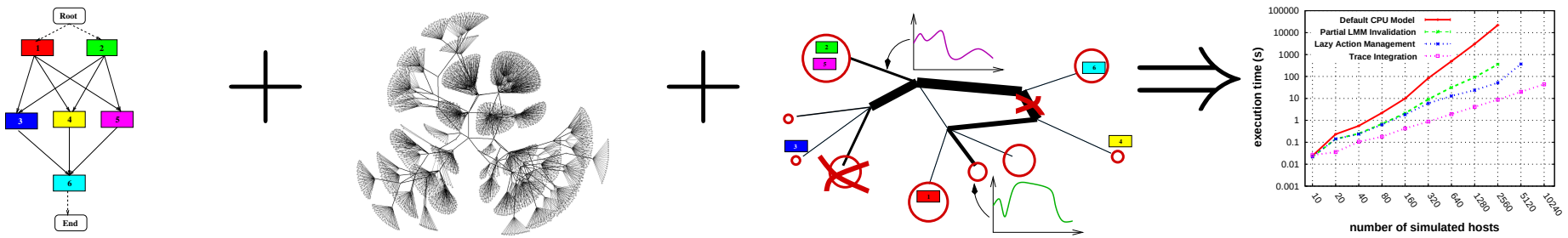
Assessing Distributed Applications' Performance

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

USS-SimGrid

Purpose of the SimGrid Project

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

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

- **Main Goal:** Make SimGrid usable in studies mandating extreme scaling

$\rightsquigarrow$ Perimeter increase from Grid Computing to Peer-to-peer

$\rightsquigarrow$ Improving simulation scalability: mandatory but not enough

$\rightsquigarrow$ Campaign data management pre- & post-processing not trivial anymore

Coming next: Some scientific achievements on these main challenges

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.

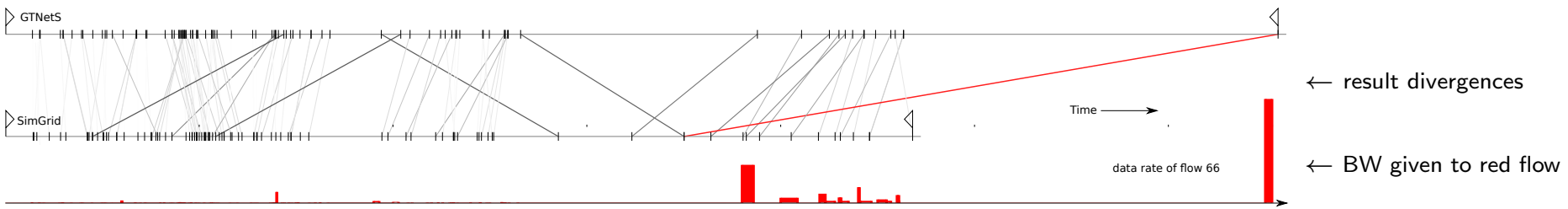
Validity: SimGrid compared to Packet-Level Tools

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

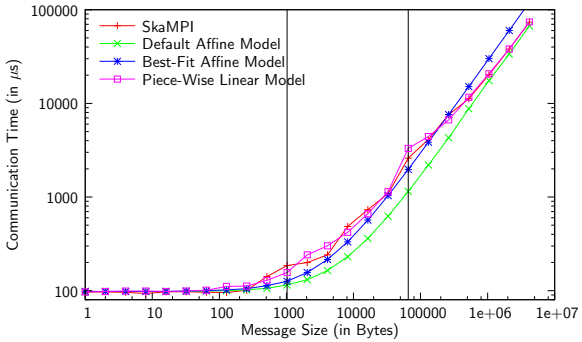


In this scenario, GTNetS and SG agree on termination date of most flows. The most diverging gets no bandwidth for a while although all others are done.

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

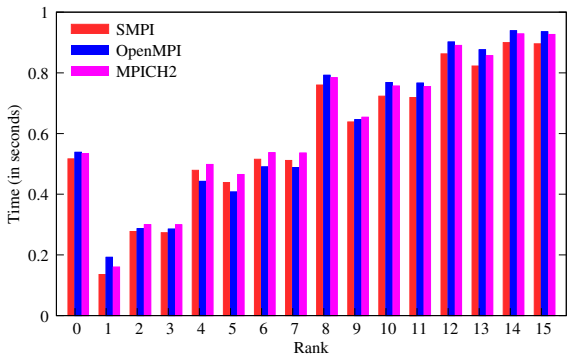
- Good prediction for short messages is crucial; Numerical instabilities deadly
- Accurately modeling MPI semantic (asynchronous & collectives ops) is tricky
- Need to account for MPI overhead; what is Real with several MPI implems?

Accuracy of MPI simulations



Timings of each communication

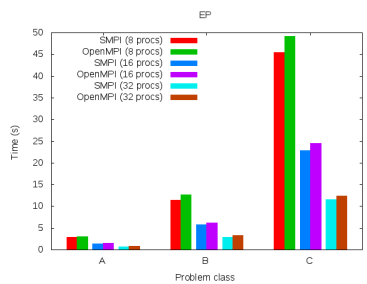
- $\lambda + size \times \tau$ not sufficient (TCP congestion)
- No affine fonction can match for all message sizes
- A 3-parts piecewise affine gives satisfying results



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

- Although not perfect, accuracy comparable / better to other MPI simulators
- *Ways* better than the most precise existing P2P simulators

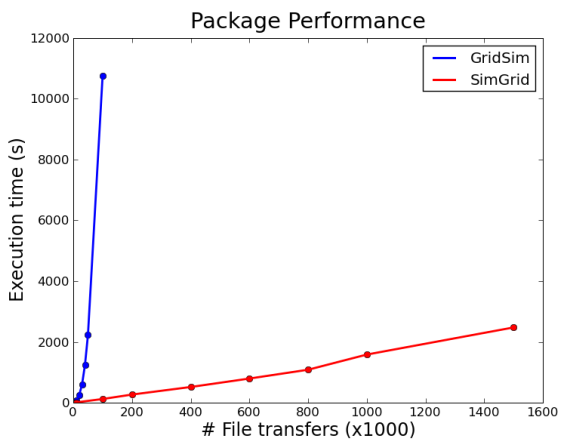
Scalability Challenge

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

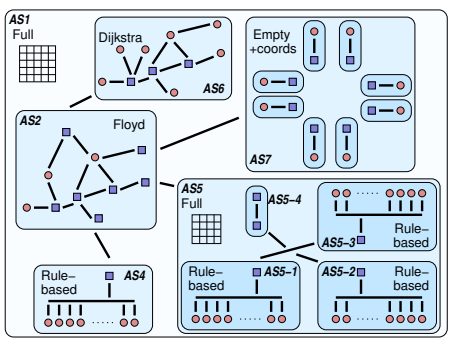
- Compact Routing Representation: From $O(n^2)$ to $O(n)$ memory consumption
- Lazy Evaluation: Arbitrary speedups on loosely coupled scenarios

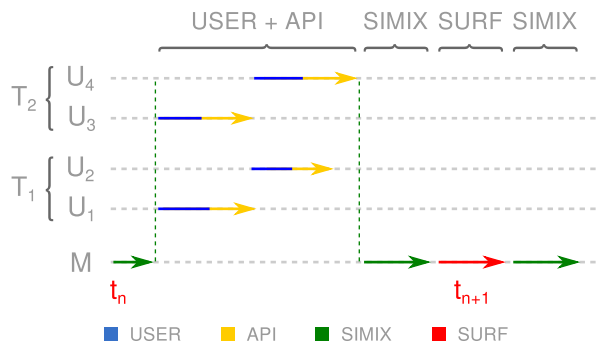
Leverage several computing units

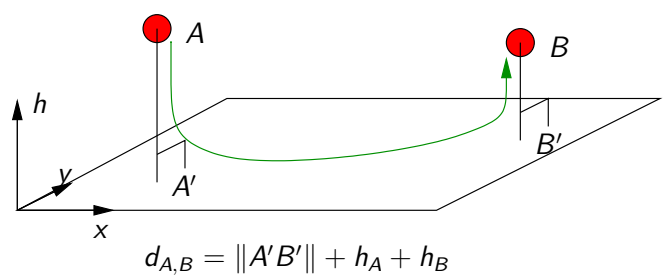
- Parallel simulation: P2P's grain so fine that classical // schema not applicable
- *Distributed simulation:* Still TBD, but not needed due to other optimizations

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



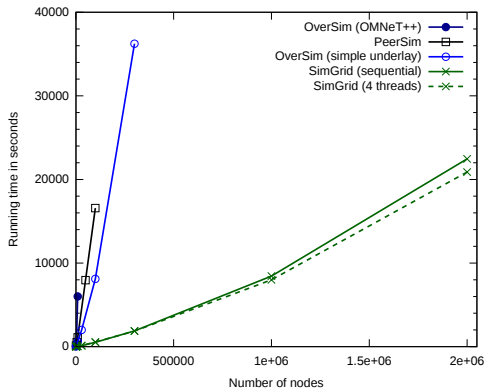




$d_{A,B} = \|A'B'\| + h_A + h_B$

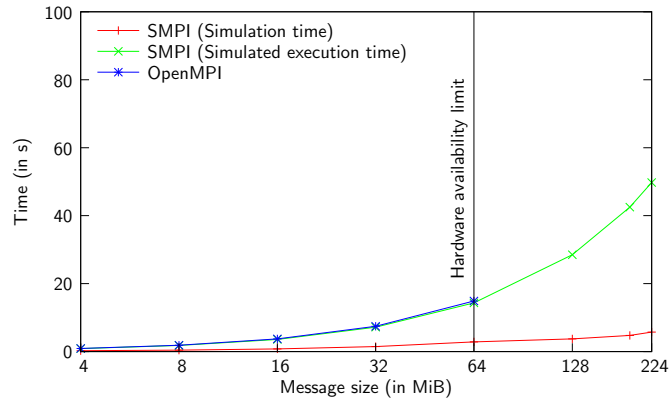
SimGrid Scalability Results

Millions of small processes



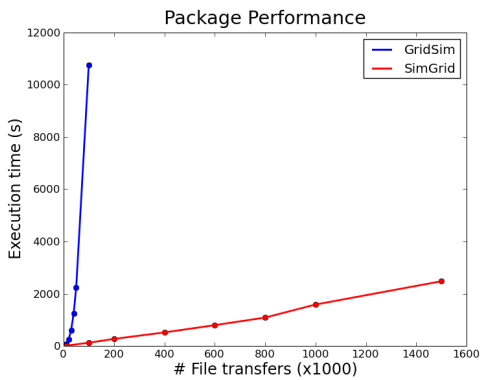
Chord simulation

Dozen of huge processes



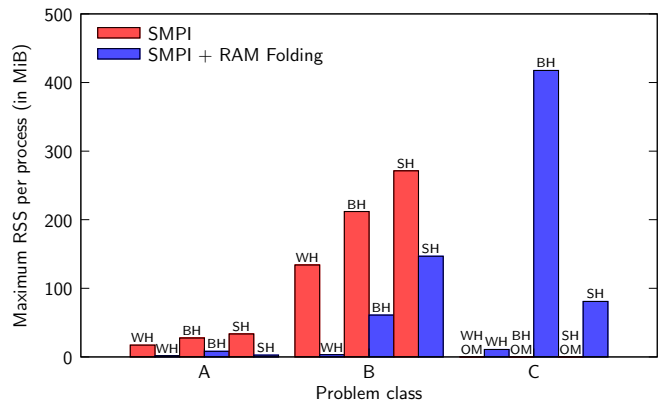
Binomial scatter with 16 processes

Large Workload



Simulation from CERN users

Hundreds of large processes



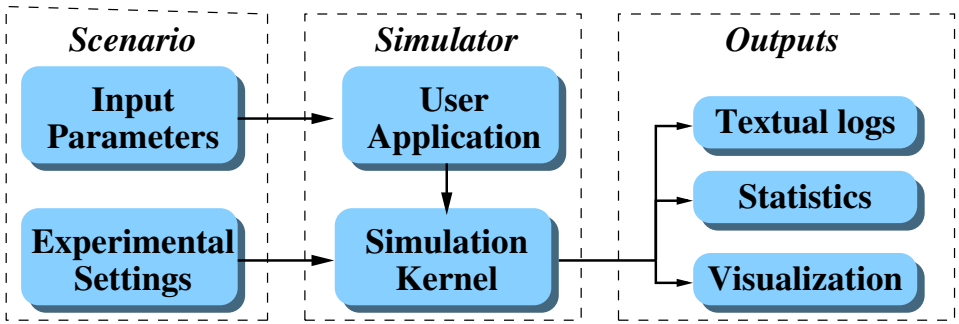
DT with up to 448 processes

Usability 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

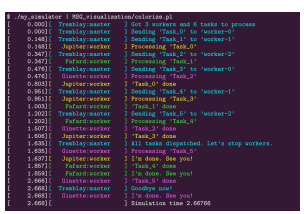


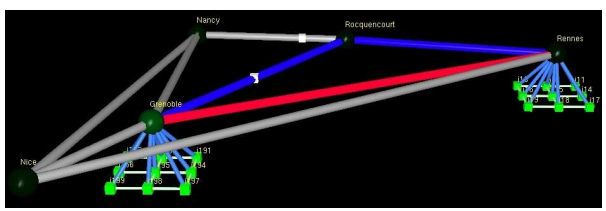
Situation before the project

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

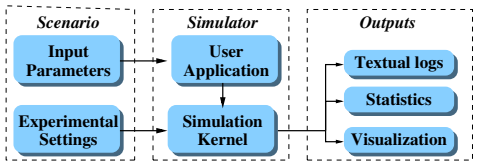
Building a *demoware* is easy. Helping understanding is harder

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



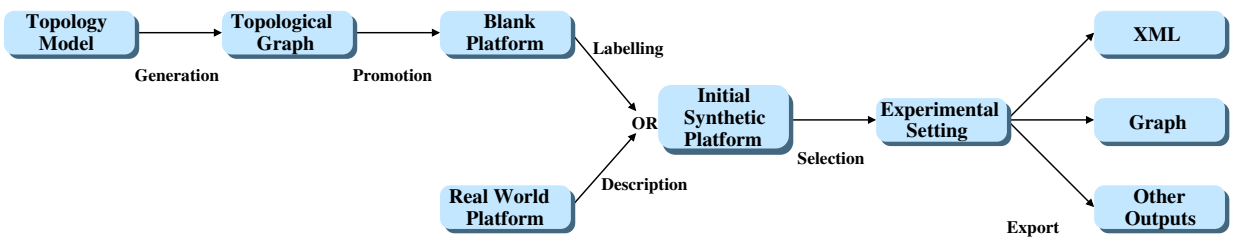


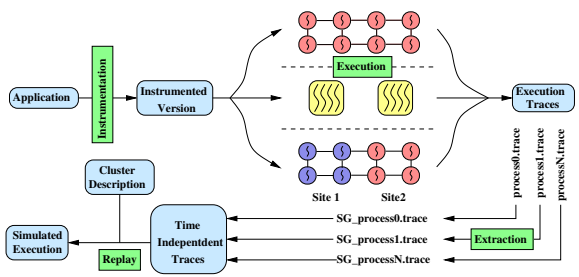
Approaches to Usability in USS-SimGrid



USS-SimGrid Proposal

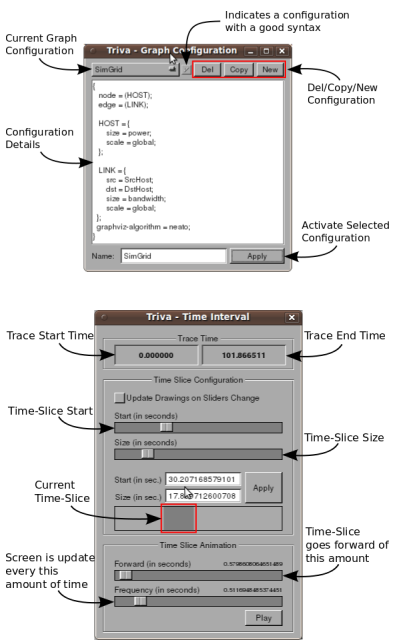
1. Workload generation:
   - Platforms: Simulacrum (generation), PDA (archive) and MintCAR (mapping)
   - Applicative Workload: Tau-based trace collection + replay
   - Background Workload: Pilgrim (trace aggregation tool)
2. Campaign management: Workflow engine
3. Single simulation analysis: Visualization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

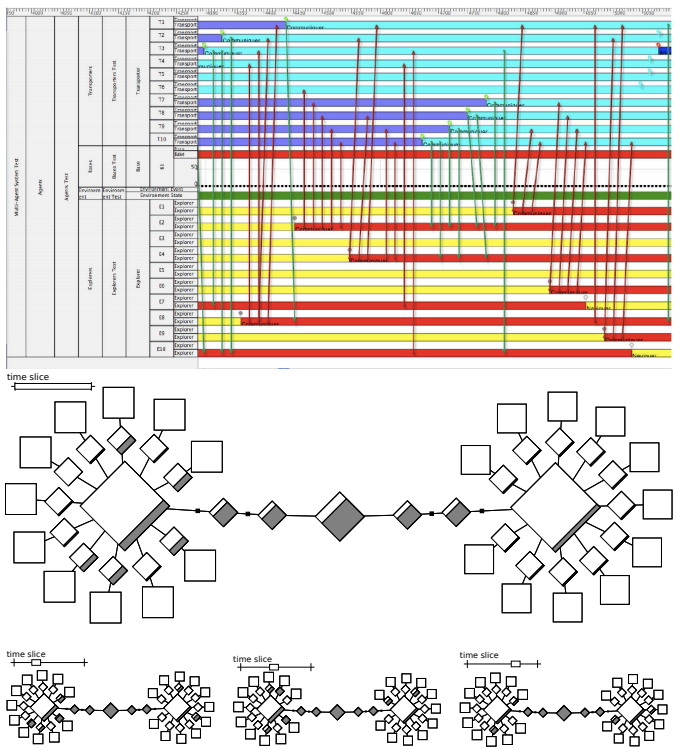


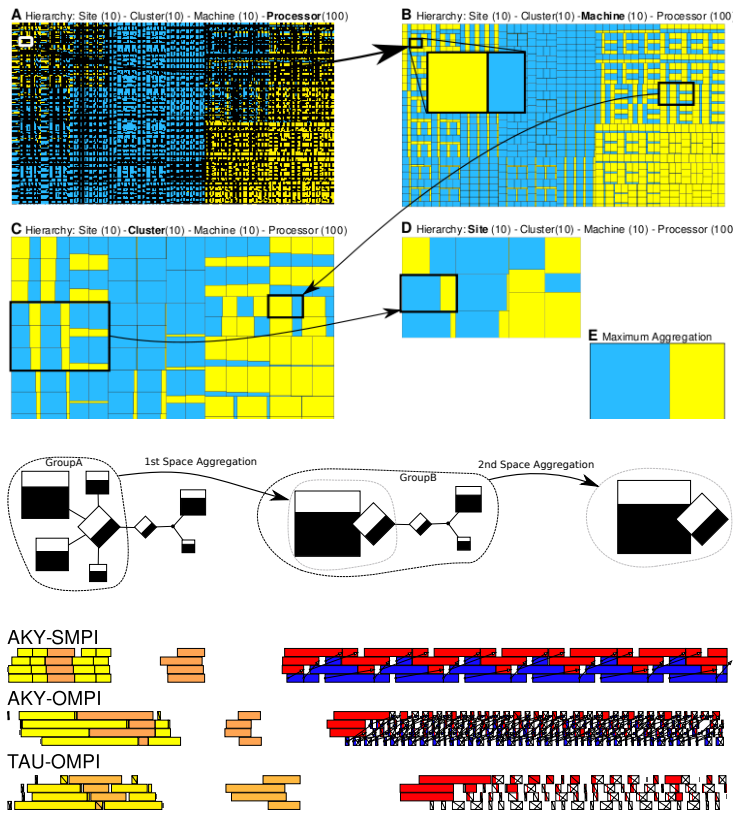


Visualizing SimGrid Simulations

- 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)







Conclusion: Project Outcomes

Scientific Production

- Publications
  - 22 international publications (including 5 multi-site publications)
  - 4 submitted articles (including 2 multi-site publications)
- Software: 11 releases of SimGrid (including 4 major releases)
  - Visualization: 4 releases of Triva, 1 release of Pajé
  - 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
- SuperComputing presence every year of project (@INRIA booth)
- 3-day Workshop: The SimGrid User Days (SUD'10)

USS-SimGrid as a Flagship (collaborative projects associated to this)

- Collaboration with ANR CIP (that use SimGrid to assess P2P HPC middleware)
- PHC Tournesol with the University of Antwerp (on scalable simulation)
- PICS CNRS Hawai'i/Villeurbanne (on MPI simulation)
- INRIA ADT (engineering forces devoted to SimGrid)

Conclusion and Open questions

Answers to good questions lead to new questions

- The work planned in this project was done on time
- But these developments gave us new ideas about going even further
- These new ideas are paving our future work

SONGS (Simulation Of Next Generation Systems) ANR project

- Making SimGrid usable in 2 more domains
  - Task 1: [Data]Grid (distributed Data mgnt for LHC; Hierarchical Storage System)
  - Task 2: Peer-to-Peer and Volunteer Computing (Replica Placement in VOD;. . .)
  - Task 3: IaaS Clouds (study from client or provider POV; energy metrics, EC2 APIs)
  - Task 4: High Performance Computing (exascale; memory & energy models)
- Further improve our Simulation Fundamentals
  - Task 5: Simulation Kernel (Efficient Simulation Kernel; DEVS Standard)
  - Task 6: Concepts and Models (energy, storage, memory, networks, volatility)
  - Task 7: Analysis and Visualization (Scalable Visualization, Causes Inference)
  - Task 8: Support to Experimental Methodology (Open Science, DoE)
- Project funded as platform project on INFRA call for 4 years (2012-2016)
- The USS adventure revealed to be the first step of the campaign. . .

Any questions?

SimGrid Internals in a Nutshell

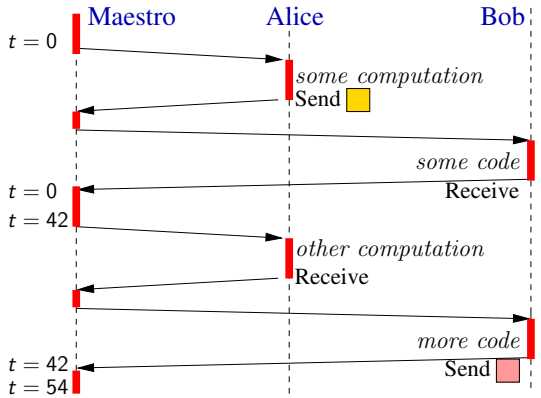
Example of user code to execute

Alice

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

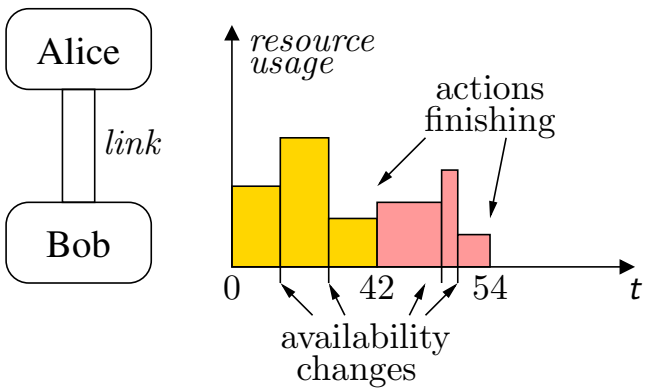
Bob

```
(some code)
Receive from Alice
(other code)
Send "blah" to Alice
```



SimGrid Internal Main Loop

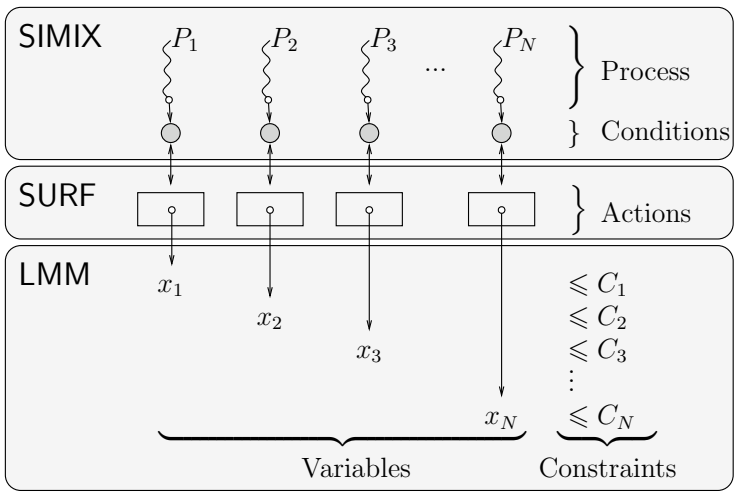
1. Run every ready user process in row
   - Each wants to consume resources
   - Assign actions on resources
2. Compute share for actions
3. Get earliest finishing action
4. Update simulated clock
5. Unlock user code waiting on this action

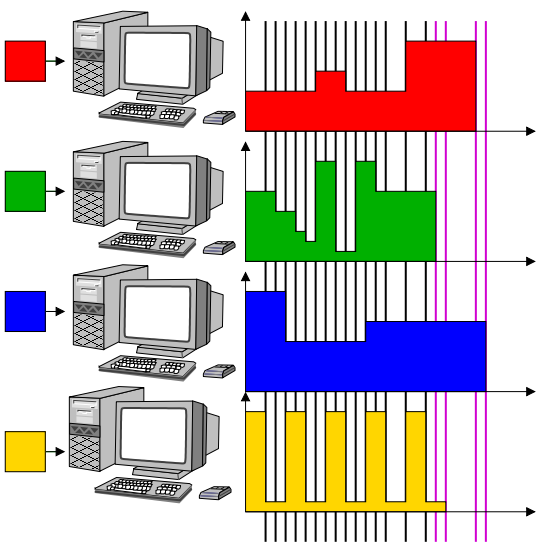


Simulation Speed Improvement (1/2)

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

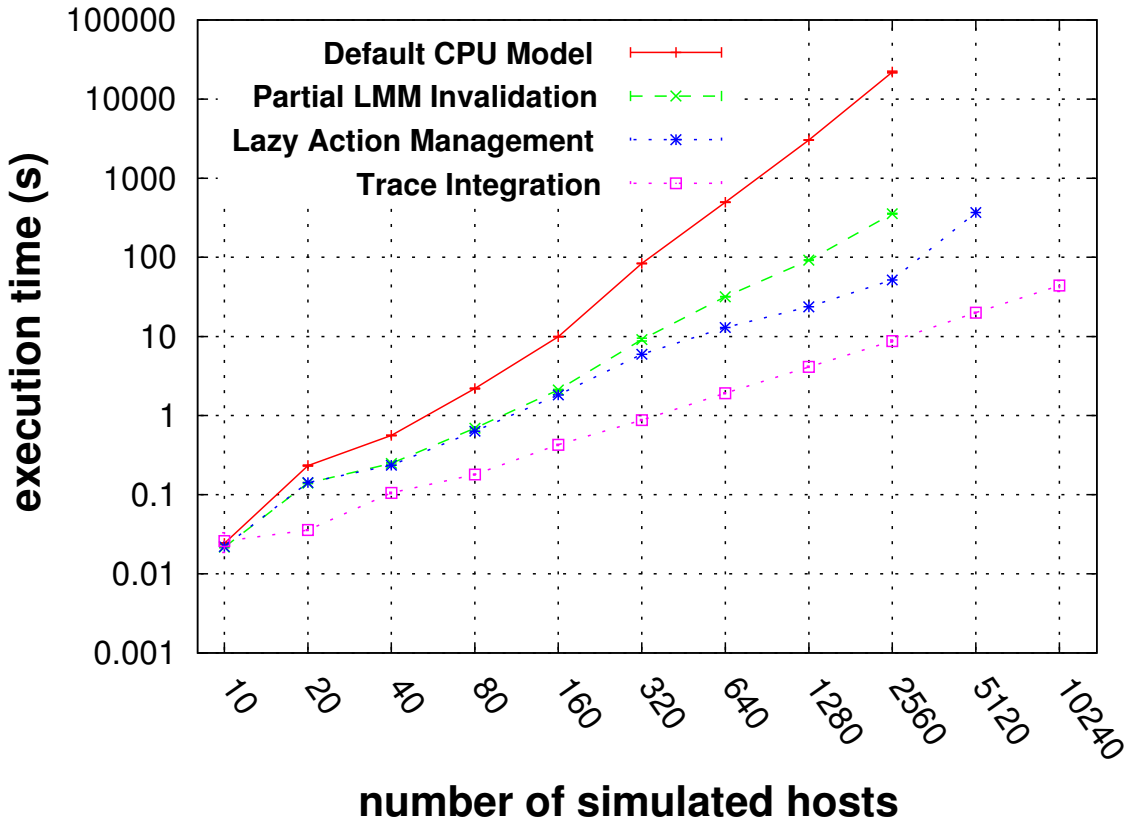
$\rightsquigarrow$ Invalidate only changed parts of the system

Availability Trace Integration

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

$\rightsquigarrow$ Precompute termination date only once

Simulation Speed Improvement (2/2)

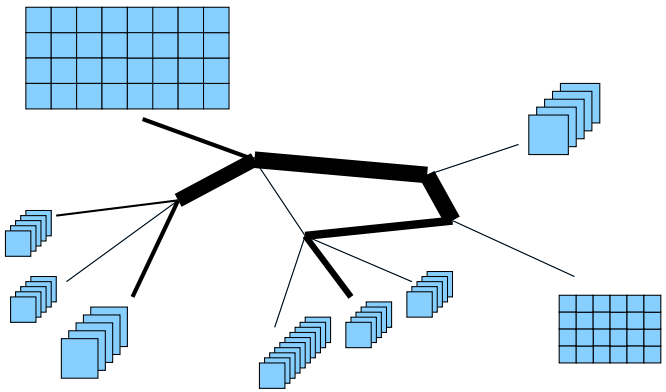


Results

- 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)

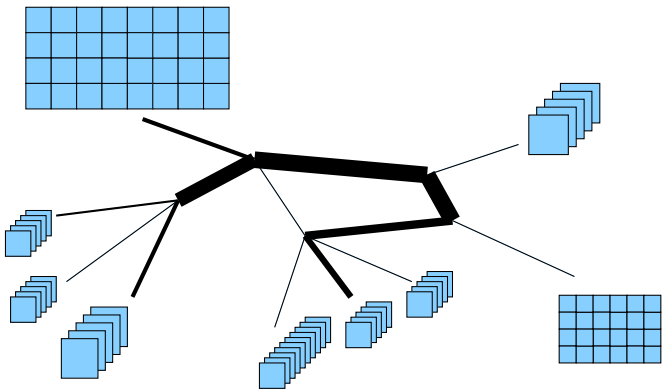
Classical Network Model in SimGrid



- 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

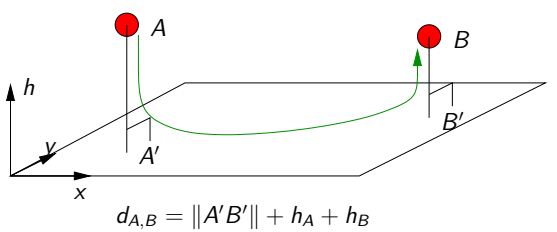


- Precise platform graph
- Needs complete routing table: quadratic size
- Limiting factor to consider larger platforms
- Acquisition/Generation is a problem
- P2P community: constant time for all coms
  Not enough info available to instanciate this

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



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

Memory Scalability: Simpler Models (2/3)

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

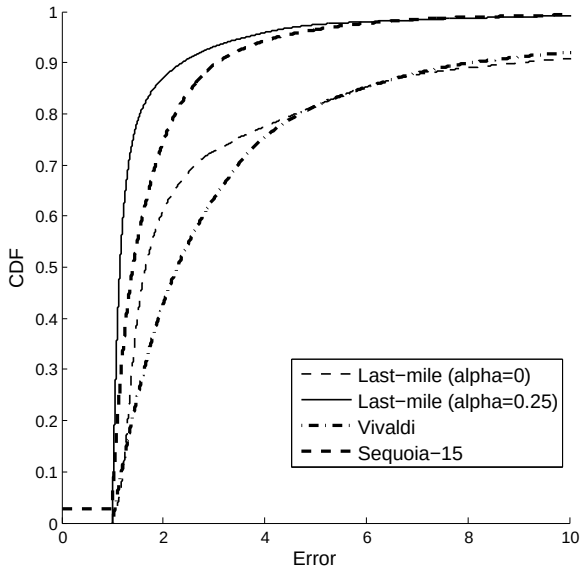
Memory Scalability: Simpler Models (3/3)

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)

Outline

- Context and Motivation
- Validity of Simulation Results
  - Context, Challenge and State of the Art
  - Validity: SimGrid compared to Packet-Level Simulators
  - Accuracy of MPI simulations
- Scalable Simulations
  - Context, Challenges, and State of the Art
  - Approaches to Scalability in USS-SimGrid
  - Results on Scalability in SimGrid
- Usability Challenge
  - Context, Challenges, and State of the Art
  - Approaches to Usability in USS-SimGrid
  - Visualizing SimGrid Simulations
- Conclusion
  - Project Outcomes
  - Conclusion and Open questions