Modéliser les systèmes à large échelle

Pourquoi? Comment?

Martin Quinson, *et Al.*
EPI Algorille – Nancy

Journées Scientifiques Inria
25 juin 2013

Contexte Scientifique

Systèmes informatiques modernes

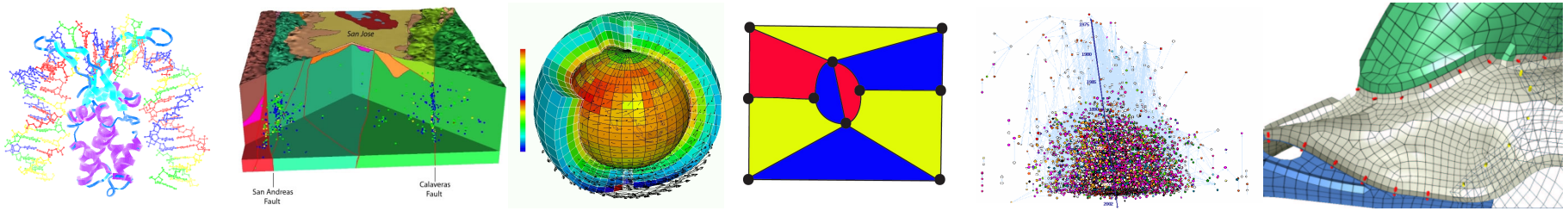
- Grilles, P2P, Clouds, HPC, ...
- **Hiérarchies** complexes et **hétérogènes**
- Systèmes **dynamiques** de très **grande taille**

Défi scientifique: **correction** et **performances** de ces systèmes

- Réductionisme insuffisant; expérience indispensable
- **Instruments scientifiques**, comme en physique ou autre

Idée: Computational Science *of* Computer Systems

- L'ordinateur comme instrument scientifique
- Des modèles pour **comprendre** et des simulation pour **prédire**



- Faire de même pour **comprendre les systèmes informatiques modernes**?

SimGrid et le projet ANR SONGS

SimGrid: Simulateur d'applications distribuées

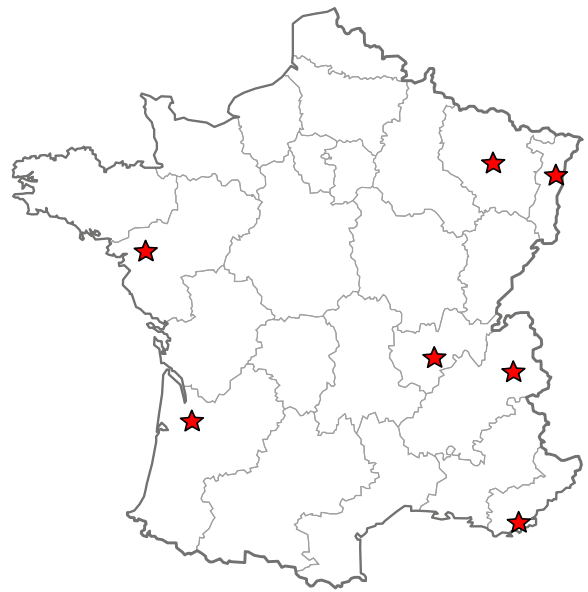
- Début (1999): Factorisation du code de quelques thésards
- Maintenant: Versatile, extensible, puissance prédictive vérifiée, libre et ouvert
- Impact (2008-2012): ≈60 publications, ≈100 auteurs, 8 EPI, 3 PhD

SONGS: Simulation Of Next Generation Systems

ANR 11 INFRA 13

- Modélisation des systèmes modernes
- Simulation de grands systèmes
- Méthodologie (planification et analyse expérimentale)

- Projet plate-forme (1.8M€), 400 PM financés
- User days et hackfest chaque année (+pléniaires)
- 7 labos partenaires, +20 chercheurs (pour 420 PM)
  (tous Inria – c'est un ANR Labs ;)



Comment modélise-t-on?

Posture épistémologique "originale"

- Complexité telle que le réductionnisme ne fonctionne plus

$$\text{Systèmes informatiques} \approx \text{Systèmes naturels}$$

- Mesures empiriques, hypothèses, modélisation, (in)validation (ad eternam)

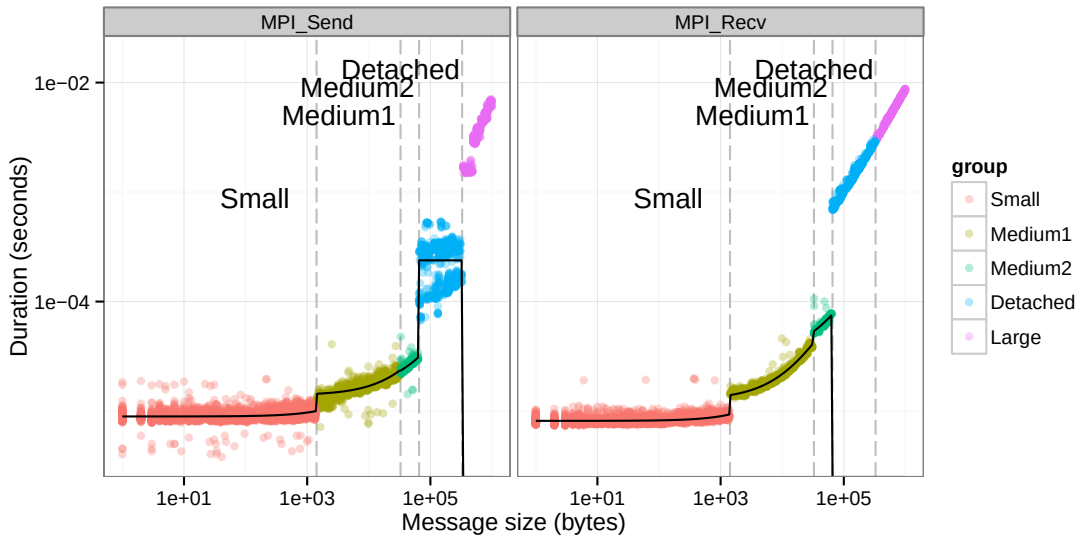
Types de modèles souhaités

- Modèles explicatifs et interprétables: On modélise surtout pour comprendre
- Quantitatif: Temps de communication ou de calcul
- Qualitatif: Interactions entre flux, ou entre processus (ou les deux)
- Sémantique: Recherche de bugs de synchronisation

Que trouve-t-on de la sorte?

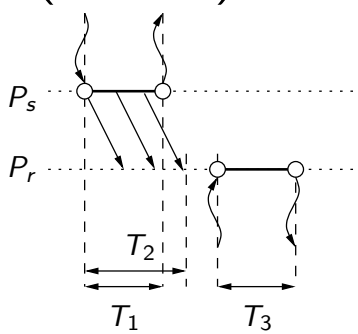
De tels modèles sont possibles

Mesures MPI_Send / MPI_Recv

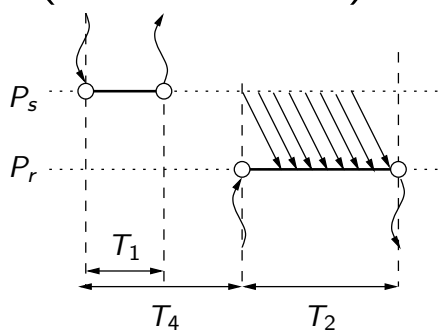


Modèles SMPI

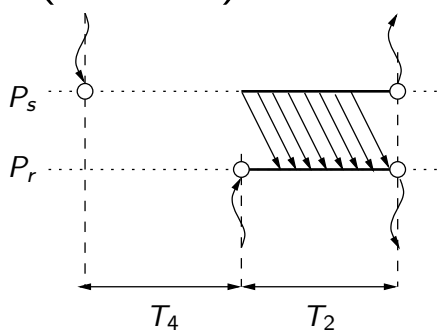
Mode asynchrone
($k \leq S_a$)



Mode détaché
($S_a < k \leq S_d$)

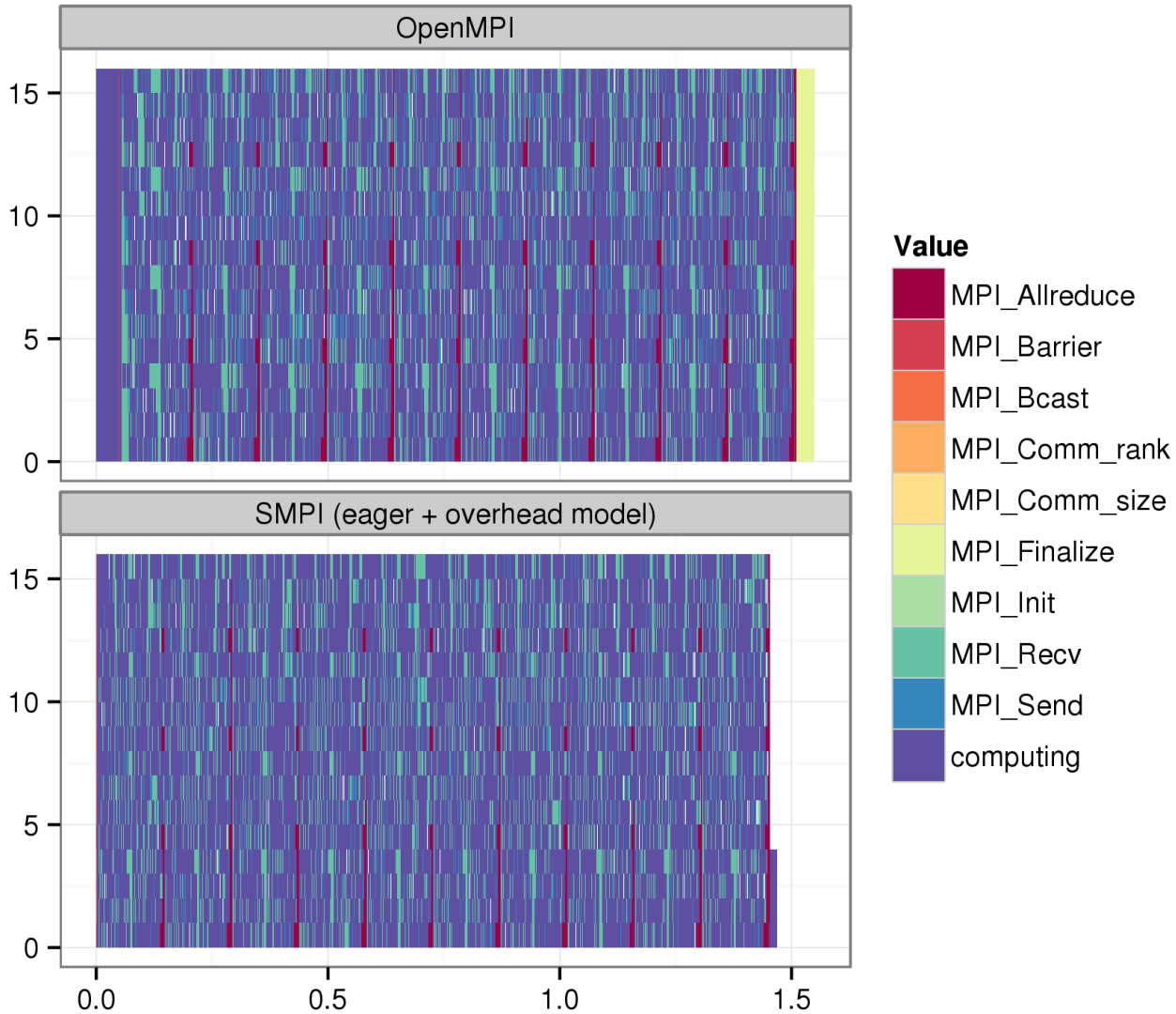


Mode synchrone
($k > S_d$)



(le modèle SimGrid capture ceci, et bien d'autres effets encore)

Et ça juste marche!

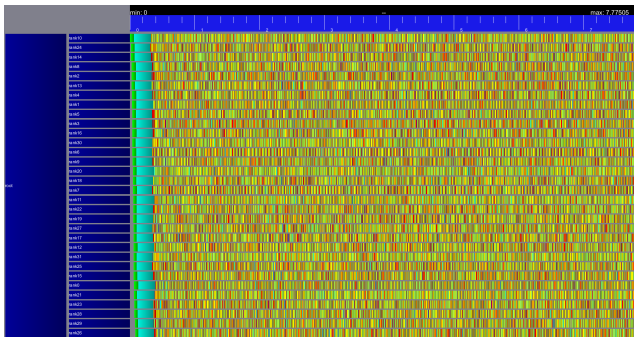


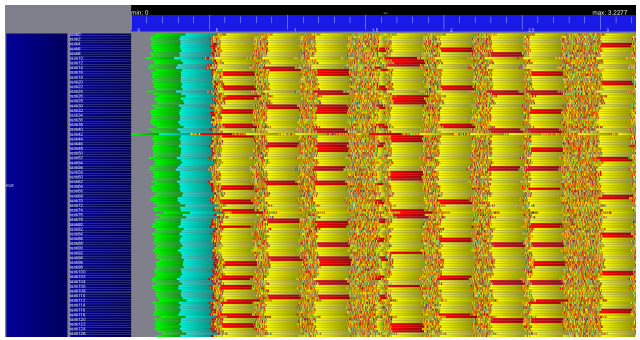
- Sweep3D: Application simple (mais non triviale) prédite dans tous ses détails
- Graphène (16 procs), OpenMPI, TCP, Gigabit Ethernet

réalisé sans overfiting :)

La réalité est souvent . . . surprenante

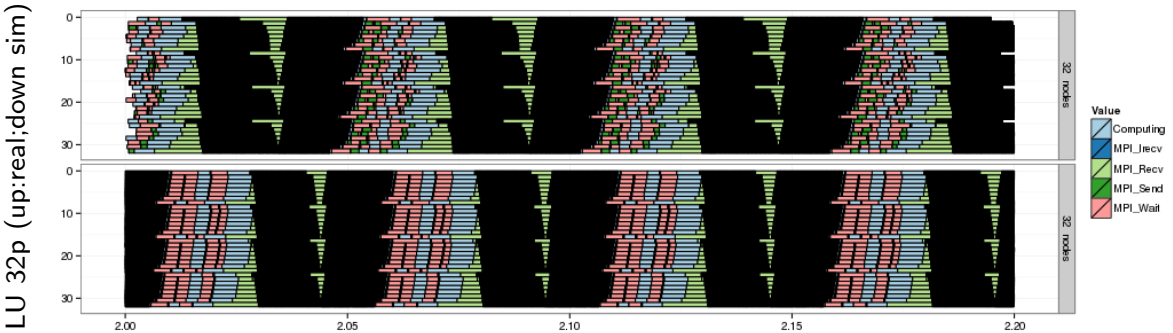
Le matériel sucks

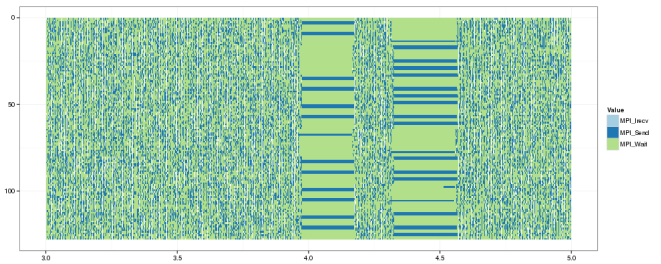




- BigDFT sur Graphene
- Bug matériel
- Paquet drops; Timeouts

TCP sucks



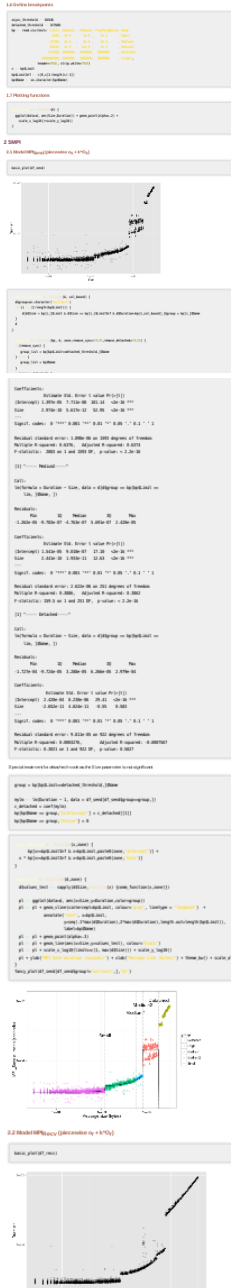


Congestion $\rightsquigarrow$ ralentissement
Vitesse $= 0 \rightsquigarrow$ timeout, et reset

On peut modéliser ces effets (et les autres)

- **Mais ne faut-il pas mieux corriger la réalité?**
- On modélisait pour comprendre ces systèmes, c'est une réussite

Autre découverte du projet: l'Open Science



Diable des détails *vs.* Graal de la reproductibilité

- Décrire l'expérience (environnement et protocole) non trivial (déluge de données)
- Expériences très sensibles: impact macro d'erreurs micro
- Post-traitement statistiques de plus en plus avancés

Mais ça aussi ça marche!

- Grid'5000 très précieux: matériel, mais aussi savoir-faire
- Nos outils (YMMV): git + org-mode + R
- Les *computational scientists* les utilisent déjà ailleurs

Reste à convaincre notre communauté ;)

- *I found the results section of this paper to be pretty weak.*
- *If less accurate models drive the user to the same conclusions (as Fig. 8 indicates), why we need more complex models?*

Conclusions

Objectif: Computational Science of Computer Systems

- Systèmes trop grands, complexes et dynamiques pour les reductionistes
- Des modèles pour comprendre; des simulations pour prédire

ANR SONGS: Simulation Of Next Generation Systems

- **Modélisation réaliste** des systèmes modernes (DataGrids, P2P, Clouds & HPC)
- Simulation extensible de grands systèmes (SimGrid as an Operating Simulator)
- Méthodologie efficace (planification, analyse expérimentale et Open Science)

Résultat 1: Des modèles suffisants pour prédire du MPI

- Il reste de nombreux points noirs, mais c'est sans précédent
- Prédictions non-triviales correctes; Réalité parfois pire que simulation :)

Résultat 2: l'Open Science ouvre un nouveau monde...

- ... où l'on comprend ce que font les autres dans leurs articles
- ... où l'on maîtrise ses propres expériences (gain de productivité)

Take Away Messages

SimGrid will prove helpful to your research

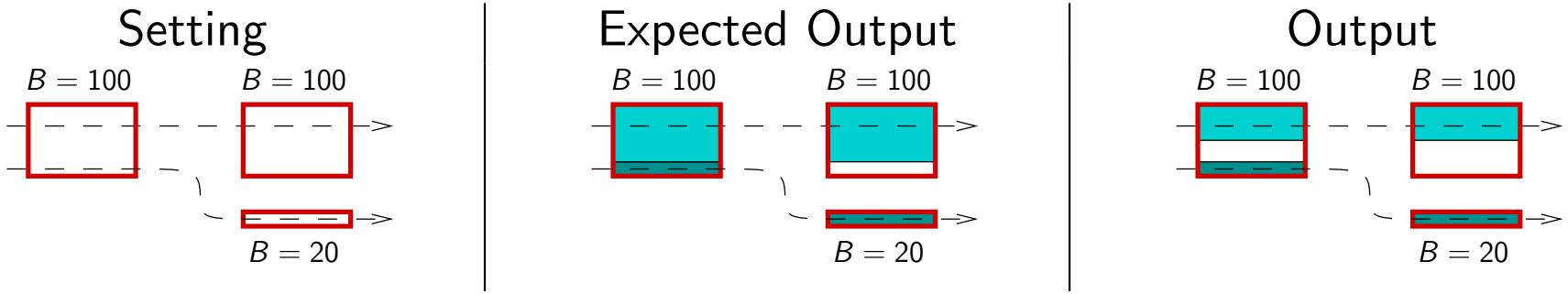
- Versatile: Used in several communities (scheduling, GridRPC, HPC, P2P, Clouds)
- Accurate: Model limits known thanks to validation studies
- Sound: Easy to use, extensible, fast to execute, scalable to death, well tested
- Open: User-community much larger than contributors group/ GPL
  120 publications (110 distinct authors, 5 continents), 4 PhD/ 25 commiters, 5 unaffiliated
- Around since over 10 years, and ready for at least 10 more years

Welcome to the Age of (Sound) Computational Science

- Discover: `http://simgrid.gforge.inria.fr/`
- Learn: 101 tutorials, user manuals and examples
- Join: user mailing list, #simgrid on irc.debian.org
  We even have some open positions ;)

Invalidating Simulators from the Litterature

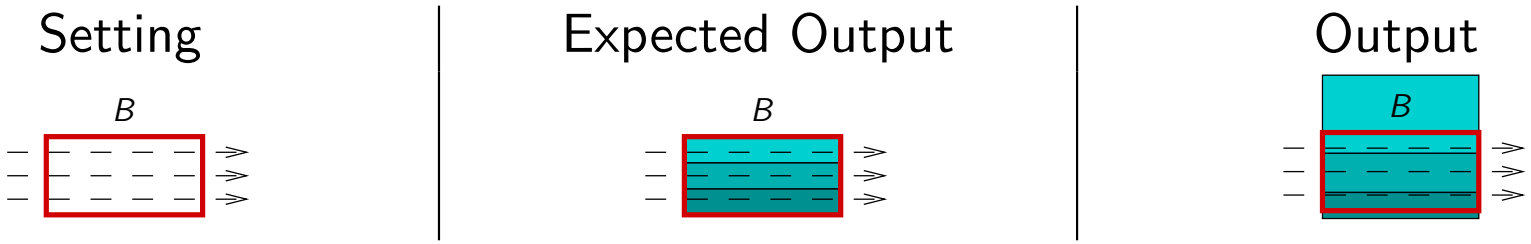
Naive flow models documented as wrong



Known issue in Narses (2002), OptorSim (2003), GroudSim (2011).

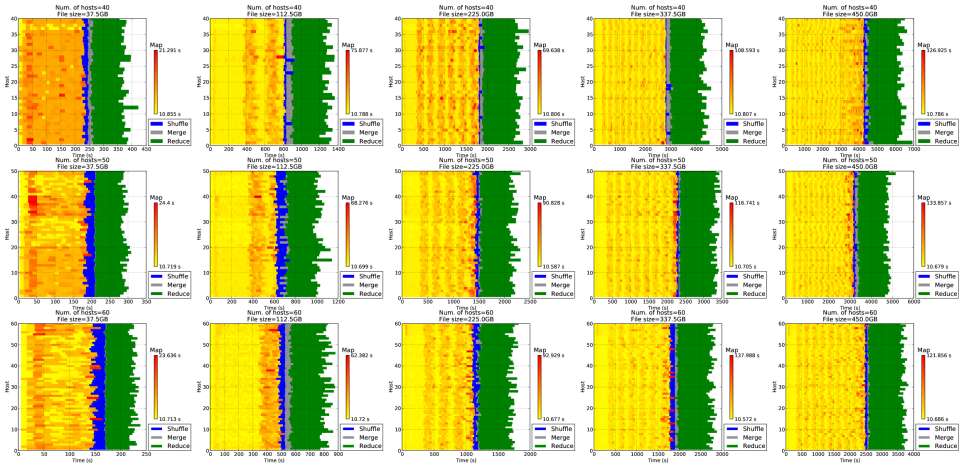
Validation by general agreement

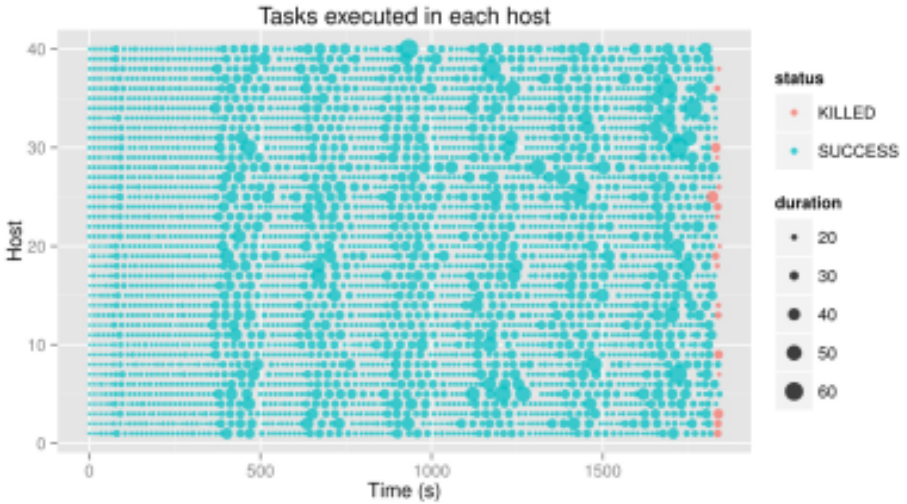
> *"Since SimJava and GridSim have been extensively utilized in conducting cutting edge research in Grid resource management by several researchers, bugs that may compromise the validity of the simulation have been already detected and fixed."* CloudSim, ICPP'09

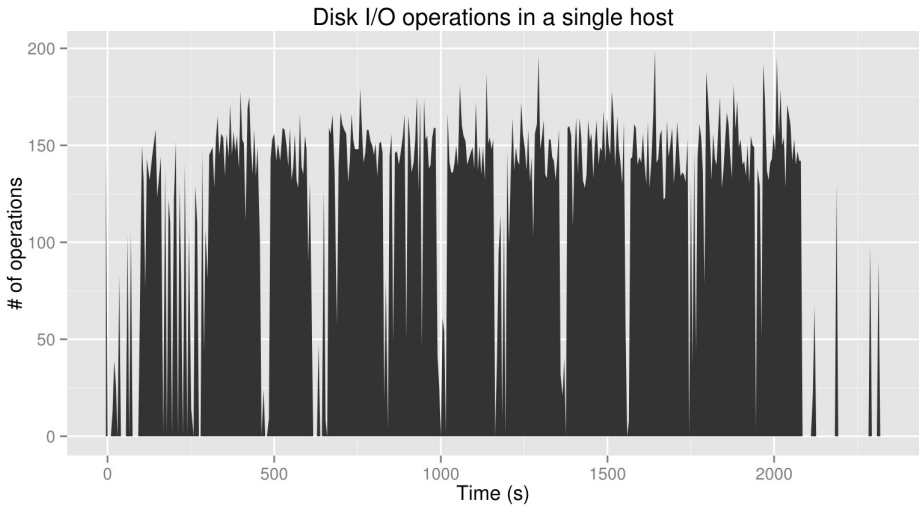


Buggy flow model (GridSim 5.2, Nov. 25, 2010). Similar issues with naive packet-level models.

MapReduce on Grid'5000







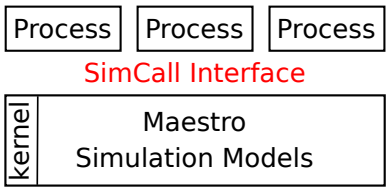
- Ralentissement CPU important
- Dû au disque IDE (pas en SATA)

Modélisable, mais faut surtout le savoir

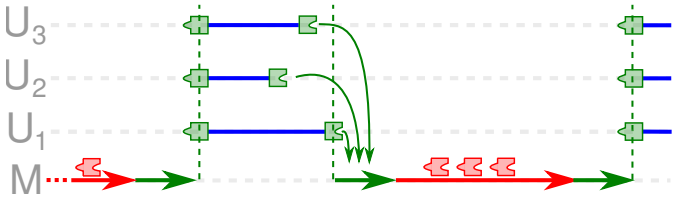
SimGrid is an Operating Simulator

OS-like internal design, isolating user processes with **simcalls**

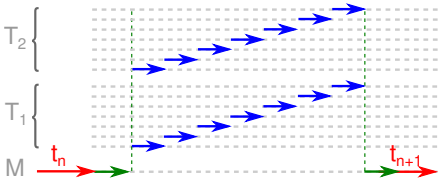
Functional View



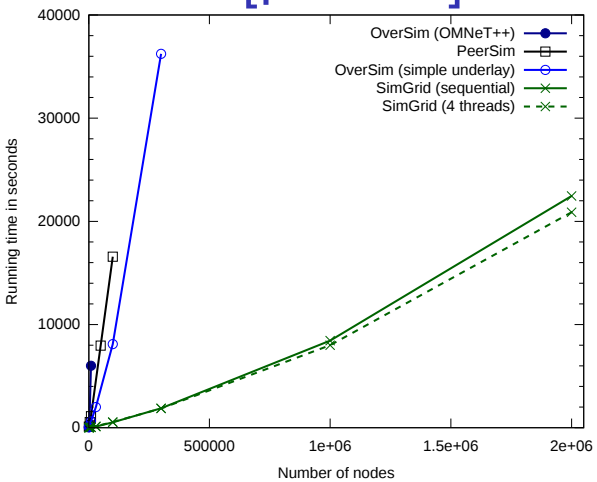
Temporal View



Implementation



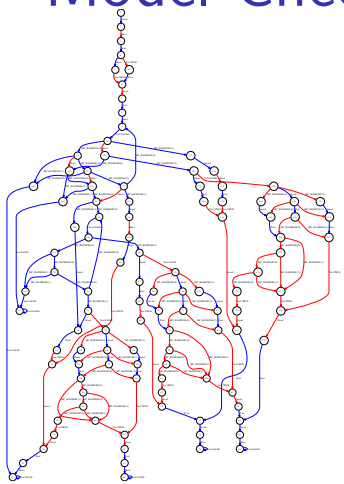
Efficient [parallel] simulation



dPeerSim: 2LP $\rightsquigarrow$ 4h / 16LP $\rightsquigarrow$ 1h

(but only 47s in sequential PeerSim, and 5s with SimGrid :)

Model-Checking (Safety & Liveness)



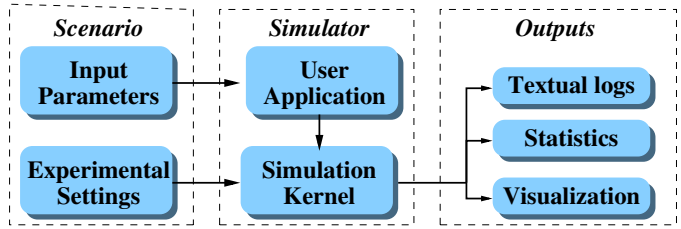
Exhaustive Chord

(2 processes)

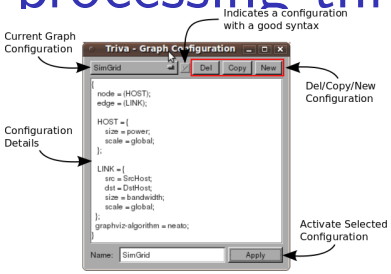
- Aims at bug finding, not assessment
- System State Equality
- + DPOR Reduction
- Soon more parallelism
- Soon statistical MC

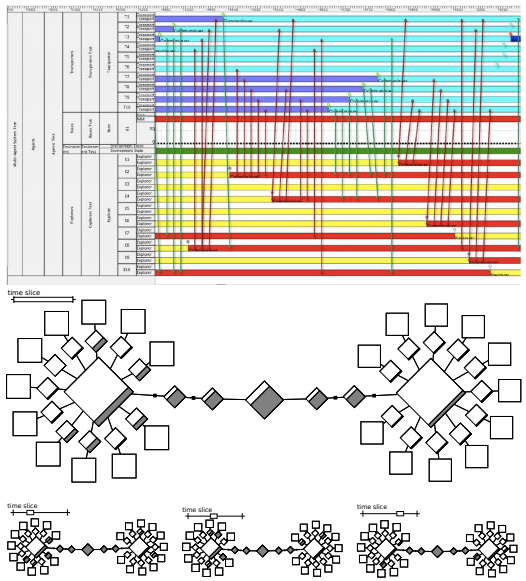
Toward an Integrated Scientific Workflow

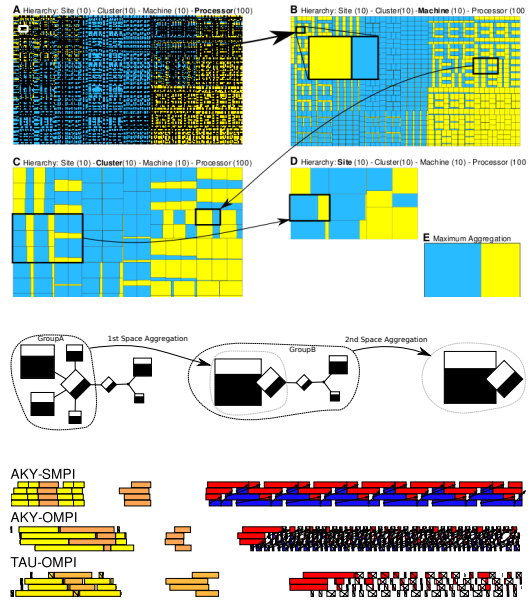
1. Prepare the experimental scenarios
2. Launch thousands of simulations
3. Post-processing and result analysis



Post-processing through Vizualization







Platform and Workload Generation