Traces de grands systèmes distribués

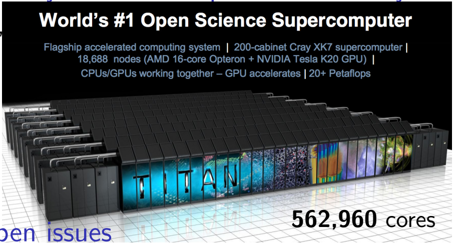
A. Legrand (MESCAL)

Séminaire Ingénierie-Recherche du LIG

Large-Scale Distributed Systems Research

Large-scale parallel and distributed systems are in production today

- HPC (clusters, petascale systems, soon exascale...)
- Grid platforms
- Peer-to-peer file sharing
- Distributed volunteer computing
- Cloud Computing



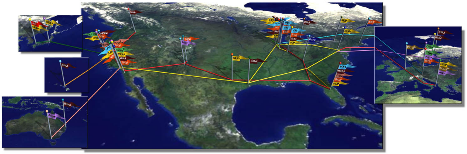
Complex platforms with many open issues

- resource discovery and monitoring
- resource & data management
- energy consumption reduction
- resource economics
- application scheduling
- fault-tolerance and availability
- scalability and performance
- decentralized algorithms

Large-Scale Distributed Systems Research

Large-scale parallel and distributed systems are in production today

- HPC (clusters, petascale systems, soon exascale...)
- Grid platforms
- Peer-to-peer file sharing
- Distributed volunteer computing
- Cloud Computing



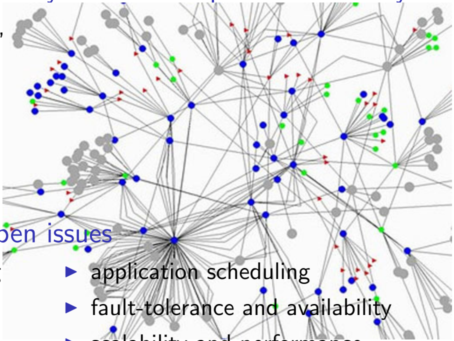
Complex platforms with many open issues

- resource discovery and monitoring
- resource & data management
- energy consumption reduction
- resource economics
- application scheduling
- fault-tolerance and availability
- scalability and performance
- decentralized algorithms

Large-Scale Distributed Systems Research

Large-scale parallel and distributed systems are in production today

- HPC (clusters, petascale systems, soon exascale...)
- Grid platforms
- Peer-to-peer file sharing
- Distributed volunteer computing
- Cloud Computing



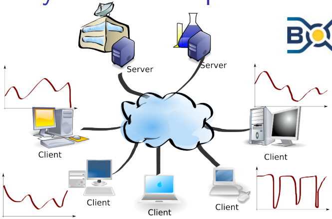
Complex platforms with many open issues

- resource discovery and monitoring
- resource & data management
- energy consumption reduction
- resource economics
- application scheduling
- fault-tolerance and availability
- scalability and performance
- decentralized algorithms

Large-Scale Distributed Systems Research

Large-scale parallel and distributed systems are in production today

- HPC (clusters, petascale systems, soon exascale...)
- Grid platforms
- Peer-to-peer file sharing
- Distributed volunteer computing
- Cloud Computing



540,000+ hosts
7.3+ PetaFLOPS

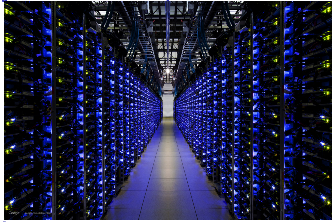
Complex platforms with many open issues

- resource discovery and monitoring
- resource & data management
- energy consumption reduction
- resource economics
- application scheduling
- fault-tolerance and availability
- scalability and performance
- decentralized algorithms

Large-Scale Distributed Systems Research

Large-scale parallel and distributed systems are in production today

- HPC (clusters, petascale systems, soon exascale...)
- Grid platforms
- Peer-to-peer file sharing
- Distributed volunteer computing
- Cloud Computing



Google Data Center

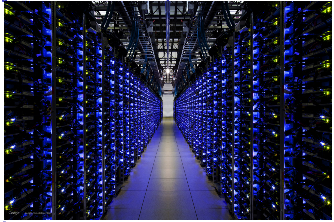
Complex platforms with many open issues

- resource discovery and monitoring
- resource & data management
- energy consumption reduction
- resource economics
- application scheduling
- fault-tolerance and availability
- scalability and performance
- decentralized algorithms

Large-Scale Distributed Systems Research

Large-scale parallel and distributed systems are in production today

- HPC (clusters, petascale systems, soon exascale...)
- Grid platforms
- Peer-to-peer file sharing
- Distributed volunteer computing
- Cloud Computing



Google Data Center

Complex platforms with many open issues

- resource discovery and monitoring
- resource & data management
- energy consumption reduction
- resource economics
- application scheduling
- fault-tolerance and availability
- scalability and performance
- decentralized algorithms

Such **applications** and **systems** deserve very advanced analysis

- Their debugging and tuning are technically difficult
- Their use induce high methodological challenges

Outline

Outline

1. Failure Trace Archive: Statistical Analysis
2. SimGrid/SMPI: HPC Application Analysis and Model Validation
3. Trace Visualization

Enabling Comparative Analysis of Diverse Distributed System

Failures in distributed systems have increasingly high negative impact and complex dynamics

Hard to evaluate and compare algorithms and models for fault-tolerance:

- Lack of public trace data sets
- Lack of standard trace format
- Lack of parsing and analytical tools

Lots of trace repository projects (PWA, Grid Observatory, GWA, ...) but little on failures.

The Failure Trace Archive: http://fta.inria.fr
(*D. Kondo, J.-M. Vincent, B. Javadi*)

- Availability traces of distributed systems, differing in scale, volatility, and usage
- Standard event-based format for failure traces
- Scripts and tools for parsing and analyzing traces in svn repository

FTA

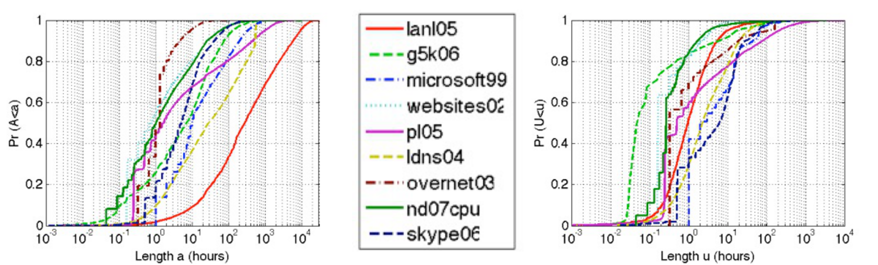
Currently 25 Data Sets including:

- SETI@home: availability of 226,208 CPU for 1.5 years (2007-2009). 2.2GB with gzip.
- Overnet: availability of 3,000 hosts was checked (probes) every 20 minutes for 2 weeks (2003)
- g5k06: availability of 2,500+ processors (Grid'5000) obtained through periodic inspection with OARmiddleware called OAR.
- microsoft99: log files of 51,663 desktops PCs at Microsoft Corporation where their reachability was determined with a ping every hour
- EGEE: state of 2500 queues for 1 month. 9.7GB with gzip

Failure modeling

Approach

- Model availability and unavailability intervals, each with a single probability distribution
- Assume availability and unavailability is identically and independently distributed
- Checking such assumptions with randomness test often leads to reject half the data



Failure modeling

Approach

- Model availability and unavailability intervals, each with a single probability distribution
- Assume availability and unavailability is identically and independently distributed
- Checking such assumptions with randomness test often leads to reject half the data

Parameter	Levels		Unit
	Low	High	
V: Volatility	$V < 50$	$V > 100$	hour
A: Availability	$A < 60$	$A > 90$	%
M: Measurement	$M < 6$	$M > 12$	month
S: Scale	$S < 1$	$S > 2$	10^3 nodes

Trace	V	A	M	S	Failure model
lanl05	L	H	H	H	Long-tailed/Long-tailed
g5k06	M	M	H	M	Long-tailed/Long-tailed
microsoft99	M	M	L	H	Short-tailed/Heavy-tailed
websites02	H	H	M	L	Long-tailed/Long-tailed
pl05	L	M	H	L	Long-tailed/Long-tailed
ldns04	L	H	L	H	Long-tailed/Short-tailed
overnet03	H	L	L	H	Short-tailed/Heavy-tailed
nd07cpu	H	M	M	L	Heavy-tailed/Long-tailed
skype06	H	L	L	H	Short-tailed/Short-tailed

Failure modeling

Approach

- Model availability and unavailability intervals, each with a single probability distribution
- Assume availability and unavailability is identically and independently distributed
- Checking such assumptions with randomness test often leads to reject half the data

For each candidate probability distribution

- Compute parameters that maximize the distribution's likelihood
- Measure goodness of fit using Kolomorov-Smirnov (KS) and Anderson-Darling (AD) tests

Other kind of study: classify by distribution (clustering)

P-Values for KS & AD Goodness-of-fit tests

p-value < 0.05 or 0.10
⇒ reject H0 that data came from fitted distribution

Availability

Trace	Exponential	Weibull	Pareto	Log-Normal	Gamma
lanl05	0.005 0.025	0.416 0.571	0.002 0.010	0.475 0.611	0.345 0.488
g5k06	0.012 0.038	0.472 0.597	0.003 0.018	0.394 0.564	0.409 0.507
microsoft99	0.005 0.084	0.294 0.546	0.000 0.049	0.371 0.611	0.198 0.418
websites02	0.000 0.006	0.079 0.354	0.000 0.027	0.188 0.401	0.055 0.182
pl05	0.000 0.000	0.080 0.245	0.002 0.016	0.168 0.321	0.043 0.131
ldns04	0.009 0.042	0.316 0.510	0.002 0.010	0.357 0.527	0.287 0.472
overnet03	0.045 0.460	0.068 0.532	0.000 0.013	0.160 0.660	0.052 0.481
nd07cpu	0.001 0.011	0.348 0.526	0.002 0.063	0.408 0.596	0.167 0.284
skype06	0.048 0.105	0.373 0.493	0.000 0.002	0.452 0.581	0.257 0.375

(Un)availability generally not heavy-tailed

Unavailability

Trace	Exponential	Weibull	Pareto	Log-Normal	Gamma
lanl05	0.000 0.004	0.196 0.346	0.000 0.001	0.481 0.607	0.042 0.095
g5k06	0.000 0.000	0.008 0.073	0.000 0.000	0.037 0.144	0.003 0.022
microsoft99	0.004 0.180	0.048 0.529	0.000 0.376	0.076 0.611	0.052 0.368
websites02	0.000 0.023	0.001 0.150	0.000 0.002	0.005 0.209	0.003 0.090
pl05	0.000 0.000	0.035 0.178	0.000 0.004	0.081 0.274	0.019 0.079
ldns04	0.035 0.112	0.404 0.538	0.000 0.001	0.464 0.607	0.277 0.411
overnet03	0.000 0.040	0.003 0.305	0.000 0.204	0.011 0.389	0.005 0.118
nd07cpu	0.000 0.004	0.028 0.219	0.000 0.031	0.126 0.559	0.003 0.032
skype06	0.071 0.191	0.288 0.478	0.002 0.015	0.182 0.449	0.267 0.408

P-Values for KS & AD Goodness-of-fit tests

p-value < 0.05 or 0.10
⇒ reject H0 that data came from fitted distribution

Availability

Trace	Exponential	Weibull	Pareto	Log-Normal	Gamma
lanl05	0.005 0.025	0.416 0.571	0.002 0.010	0.475 0.611	0.345 0.488
g5k06	0.012 0.038	0.472 0.597	0.003 0.018	0.394 0.564	0.409 0.507
microsoft99	0.005 0.084	0.294 0.546	0.000 0.049	0.371 0.611	0.198 0.418
websites02	0.000 0.006	0.079 0.354	0.000 0.027	0.188 0.401	0.055 0.182
pl05	0.000 0.000	0.080 0.245	0.002 0.016	0.168 0.321	0.043 0.131
ldns04	0.009 0.042	0.316 0.510	0.002 0.010	0.357 0.527	0.287 0.472
overnet03	0.045 0.460	0.068 0.532	0.000 0.013	0.160 0.660	0.052 0.481
nd07cpu	0.001 0.011	0.348 0.526	0.002 0.063	0.408 0.596	0.167 0.284
skype06	0.048 0.105	0.373 0.493	0.000 0.002	0.452 0.581	0.257 0.375

Unavailability

Trace	Exponential	Weibull	Pareto	Log-Normal	Gamma
lanl05	0.000 0.004	0.196 0.346	0.000 0.001	0.481 0.607	0.042 0.095
g5k06	0.000 0.000	0.008 0.073	0.000 0.000	0.037 0.144	0.003 0.022
microsoft99	0.004 0.180	0.048 0.529	0.000 0.376	0.076 0.611	0.052 0.368
websites02	0.000 0.023	0.001 0.150	0.000 0.002	0.005 0.209	0.003 0.090
pl05	0.000 0.000	0.035 0.178	0.000 0.004	0.081 0.274	0.019 0.079
ldns04	0.035 0.112	0.404 0.538	0.000 0.001	0.464 0.607	0.277 0.411
overnet03	0.000 0.040	0.003 0.305	0.000 0.204	0.011 0.389	0.005 0.118
nd07cpu	0.000 0.004	0.028 0.219	0.000 0.031	0.126 0.559	0.003 0.032
skype06	0.071 0.191	0.288 0.478	0.002 0.015	0.182 0.449	0.267 0.408

Gamma a good fit. Amenable for Markov Models

P-Values for KS & AD Goodness-of-fit tests

p-value < 0.05 or 0.10
⇒ reject H0 that data came from fitted distribution

Availability

Trace	Exponential	Weibull	Pareto	Log-Normal	Gamma
lanl05	0.005 0.025	0.416 0.571	0.002 0.010	0.475 0.611	0.345 0.488
g5k06	0.012 0.038	0.472 0.597	0.003 0.018	0.394 0.564	0.409 0.507
microsoft99	0.005 0.084	0.294 0.546	0.000 0.049	0.371 0.611	0.198 0.418
websites02	0.000 0.006	0.079 0.354	0.000 0.027	0.188 0.401	0.055 0.182
pl05	0.000 0.000	0.080 0.245	0.002 0.016	0.168 0.321	0.043 0.131
ldns04	0.009 0.042	0.316 0.510	0.002 0.010	0.357 0.527	0.287 0.472
overnet03	0.045 0.460	0.068 0.532	0.000 0.013	0.160 0.660	0.052 0.481
nd07cpu	0.001 0.011	0.348 0.526	0.002 0.063	0.408 0.596	0.167 0.284
skype06	0.048 0.105	0.373 0.493	0.000 0.002	0.452 0.581	0.257 0.375

Unavailability

Trace	Exponential	Weibull	Pareto	Log-Normal	Gamma
lanl05	0.000 0.004	0.196 0.346	0.000 0.001	0.481 0.607	0.042 0.095
g5k06	0.000 0.000	0.008 0.073	0.000 0.000	0.037 0.144	0.003 0.022
microsoft99	0.004 0.180	0.048 0.529	0.000 0.376	0.076 0.611	0.052 0.368
websites02	0.000 0.023	0.001 0.150	0.000 0.002	0.005 0.209	0.003 0.090
pl05	0.000 0.000	0.035 0.178	0.000 0.004	0.081 0.274	0.019 0.079
ldns04	0.035 0.112	0.404 0.538	0.000 0.001	0.464 0.607	0.277 0.411
overnet03	0.000 0.040	0.003 0.305	0.000 0.204	0.011 0.389	0.005 0.118
nd07cpu	0.000 0.004	0.028 0.219	0.000 0.031	0.126 0.559	0.003 0.032
skype06	0.071 0.191	0.288 0.478	0.002 0.015	0.182 0.449	0.267 0.408

Weibull and Log Normal provide best fit

Outline

1. Failure Trace Archive: Statistical Analysis
2. SimGrid/SMPI: HPC Application Analysis and Model Validation
3. Trace Visualization

SimGrid

MPI simulation: what for ?

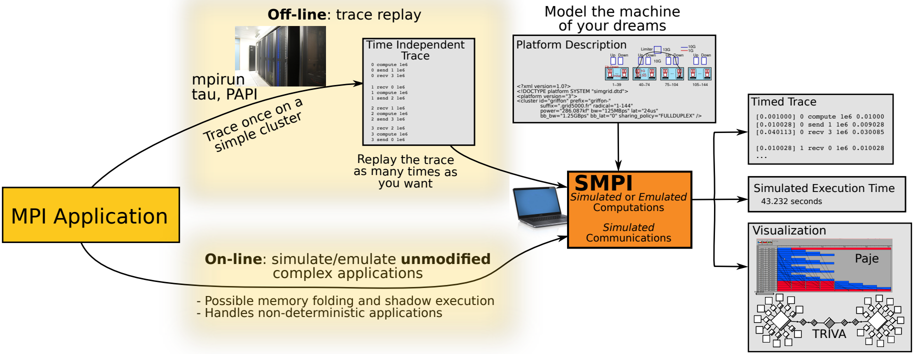
1. Helping application developers
   - Non-intrusive tracing and repeatable execution
   - Classical debugging tools (gdb, valgrind) can be used
   - Save computing resources (runs on your laptop if possible)
2. Helping application users (provide sound baseline)
3. Capacity planning (can we save on components? what-if analysis)

Simulation can help but **packet-level simulations are not an option**.

SimGrid in a nutshell

- A 13 years old, open source/science project (France, USA, . . .)
- SimGrid relies on flow-level models that take topology into account.
  - Many naive flow-level models implemented in other simulators are *documented as wrong*
  - Some tools are *validated by general agreement*
  - Some tools present convincing graphs, which are *hardly reproducible*
  - Some tools are *optimistically validated*
  - Instead, we try to invalidate and improve our models

SMPI – Offline vs. Online Simulation

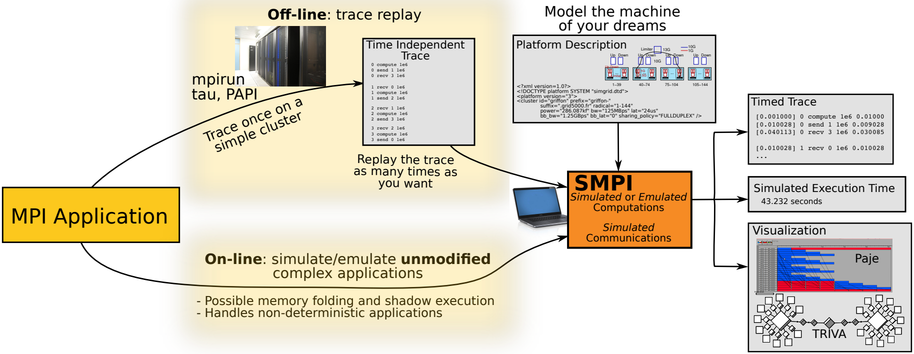


Offline simulation

1. Obtain a time independent trace
2. Replay it on top of SimGrid as often as desired
3. Analyze with the comfort of a simulator

Fast, but requires extrapolation and limited to non-adaptive codes

SMPI – Offline vs. Online Simulation



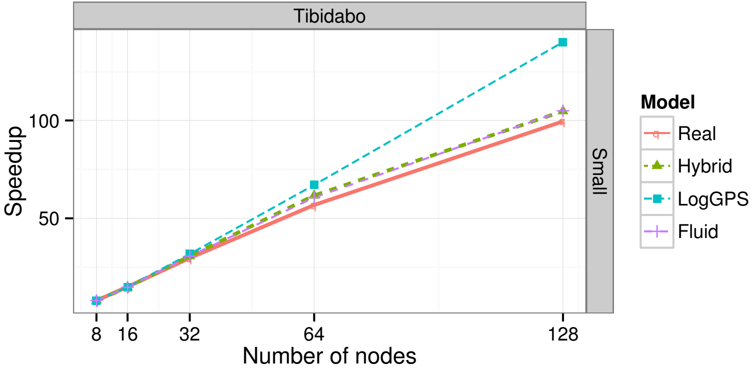
Online simulation

- Directly run the code on top of SimGrid
- Possible memory sharing between simulated processes (reduces memory footprint) and kernel sampling (reduces simulation time)
- Complies with most of the MPICH3 testsuite, compatible with many C F77 and F90 codes (NAS, LinPACK, Sweep3D, BigDFT, SpecFEM3D)

Validation: Non-trivial Application Scaling

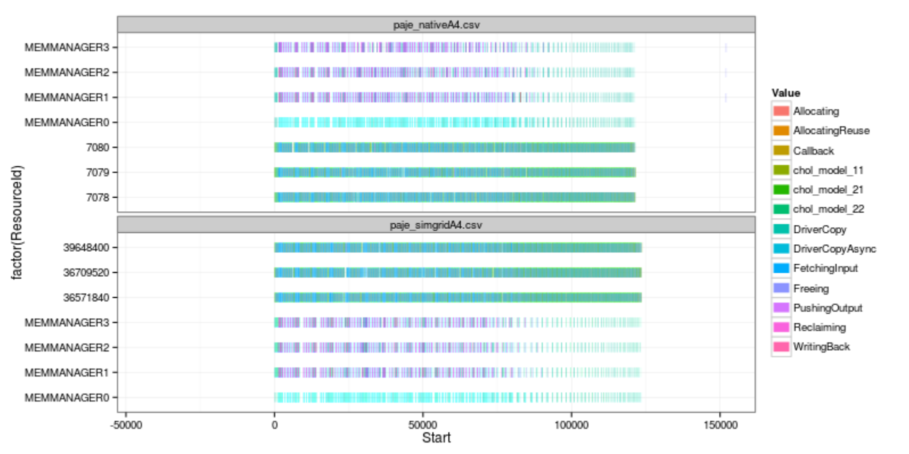
Experiments also run using real Physics code (BigDFT, SPECFEM3D) on Tibidabo (ARM cluster prototype)

- The set of collective operations may completely change depending on the instance, hence the need to use online simulation
- Very good accuracy (especially compared to LogP)



Similar Related Ongoing Work: StarPU

StarPU : a Dynamic Task-Based Runtime System for Heterogeneous Multi-Core Architectures (Bordeaux)



A few figures

SMPI article

- http://hal.inria.fr/hal-00919507
- 26GB of uncompressed traces
- Available on Figshare *a posteriori*
  http://figshare.com/articles/SC13_Publication_on_SMPI/833851
- Store the org-file that combines the article and all the scripts

StarPU study

- This time, traces are uploaded while conducting the study
  http://figshare.com/account/projects/326 *a minima*
- Keep using org-mode and R for writing reproducible articles

Outline

Data Aggregation and Alternative Visualization Techniques for Parallel and Distributed Program Analysis

Lucas Mello Schnorr (CNRS)
LIG-MESCAL, Grenoble, France

TUD-ZIH-Colloquium – Dresden, Germany

July 26th, 2012

Courtesy of Lucas Schnorr

Visualizing More Performance Data Than What Fits on Your Screen

Challenges and Motivation

- Very large applications
  → Top500 has machines with 1.5 million cores
- Low-intrusion tracing techniques
  → Buffering, hardware support, simulation traces

Space/Time trace size explosion

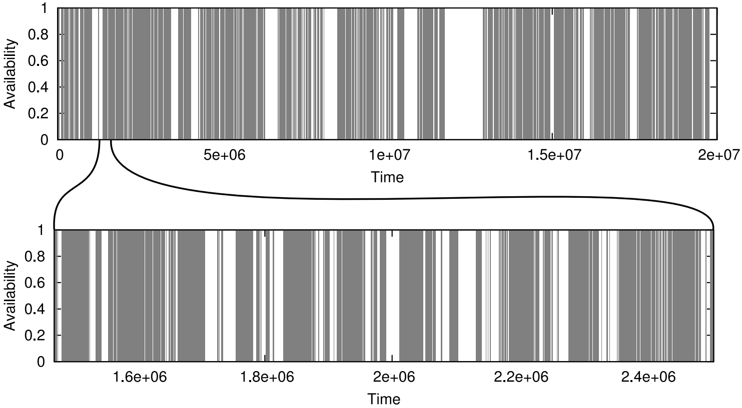
- Very detailed in time, many entities in space
- Data representation without care
  → may deceive understanding

- Real BOINC availability trace file
  - Availability is either true or false
  - 8-month period, then 12-day zoom
  - One volunteer
- Plot with GNUPlot to a PDF (vector) file

Courtesy of Lucas Schnorr

Motivation (BOINC example)

- One volunteer client (top: 8-month, bottom: 12-day)
- Reasonable view, with a zoom for details



Courtesy of Lucas Schnorr

Motivation – trust the rendering?

Same vector file, two different views

→ Different interpretation depending on the viewer

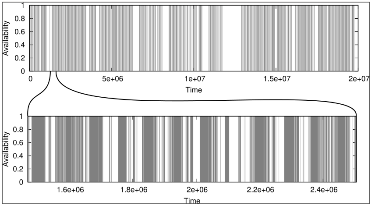
Courtesy of Lucas Schnorr

Motivation – trust the rendering?

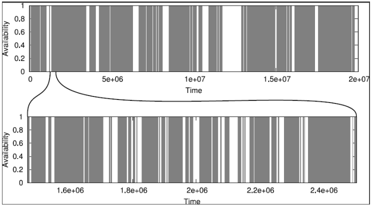
Same vector file, two different views

$\rightarrow$ Different interpretation depending on the viewer

Evince



Acroread



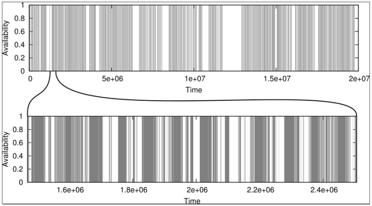
Courtesy of Lucas Schnorr

Motivation – trust the rendering?

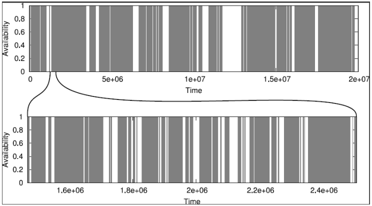
Same vector file, two different views

$\rightarrow$ Different interpretation depending on the viewer

Evince



Acroread

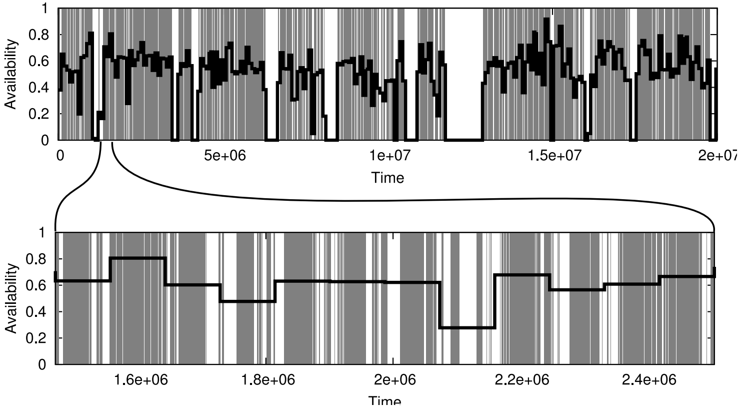


- Should we trust the rendering ?
  - No!
  - We need to make choices before visualizing data

Courtesy of Lucas Schnorr

Motivation → data aggregation

- 24-hour time integration

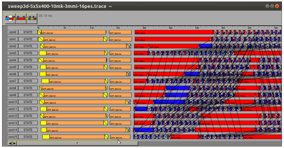


Courtesy of Lucas Schnorr

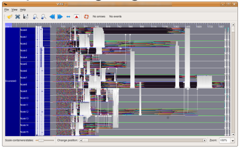
Space/Time views for trace analysis

- Widespread, useful, intuitive, fast adoption
- Space (vertical axis) and Time (horizontal)
- All trace events represented, causal order

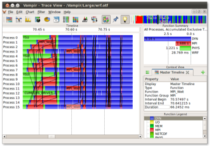
Paje
http://paje.sf.net



Vite
http://vite.gforge.inria.fr



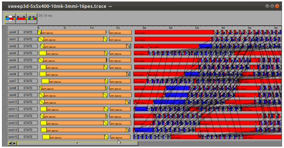
Vampir
http://vampir.eu



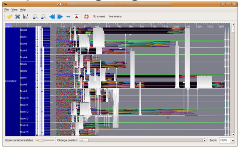
Space/Time views for trace analysis

- Widespread, useful, intuitive, fast adoption
- Space (vertical axis) and Time (horizontal)
- All trace events represented, causal order

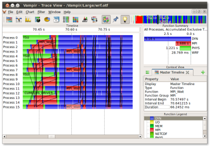
Paje
http://paje.sf.net



Vite
http://vite.gforge.inria.fr



Vampir
http://vampir.eu



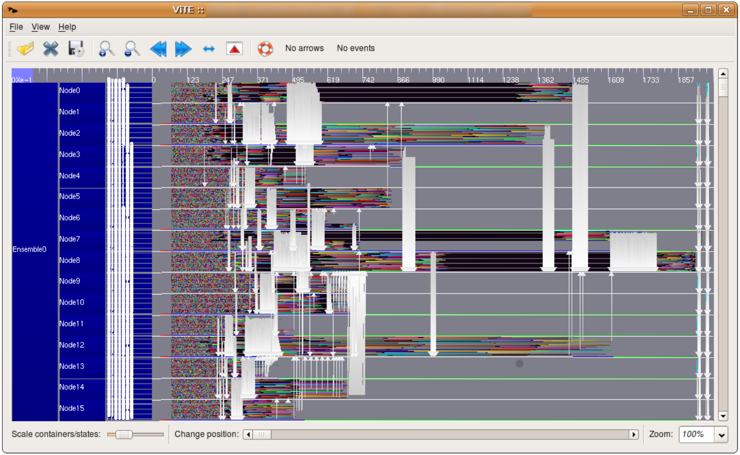
However...

- Also impacted by ever larger trace sizes
- Limited visualization scalability

Courtesy of Lucas Schnorr

Space/Time views – closer look (ViTe tool)

- Trust the OpenGL rendering, no data aggregation

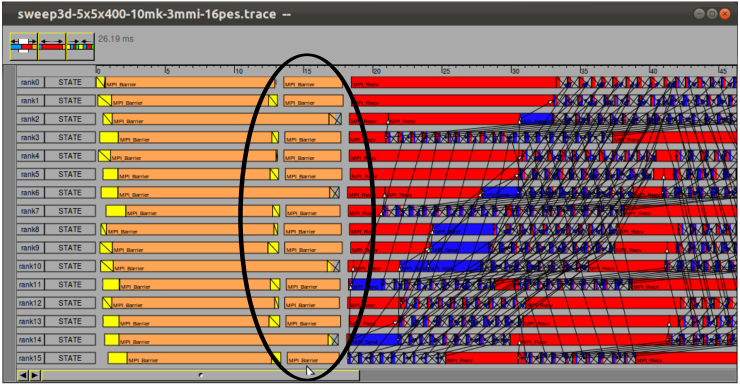


Source: http://vite.gforge.inria.fr

Courtesy of Lucas Schnorr

Space/Time views – closer look (old Pajé)

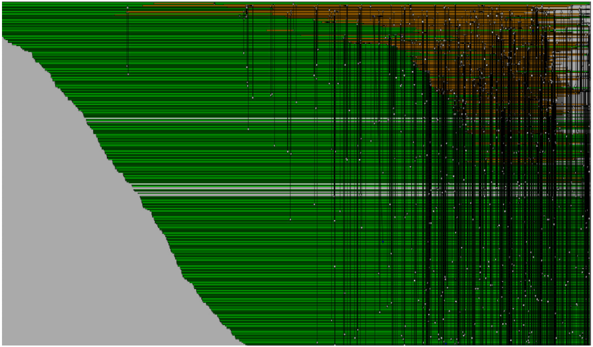
- Opaque aggregating filter (no user interaction)
  → Slashed rectangles represent time-integrated states
- Self-configure depending on temporal zoom



Source: `http://paje.sourceforge.net`

Space/Time views – closer look (old Pajé)

- Space dimension: one process per vertical pixel
  → at best, 1000 process represented at the same time

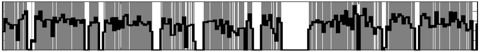


- Clustering algorithms by process behavior?
  → Remove similar processes and choose a representative

Courtesy of Lucas Schnorr

Introduction – summary and approach

- Data aggregation is key for large-scale visualization
  $\rightarrow$ Avoid graphical aggregation rendering
- Aggregated data may be more representative



- Note: Concerns with behavior attenuation
  - Aggregation may remove important details
  - Flexible aggregation: operators & neighborhood

Main idea: Visualization techniques based upon aggregated data

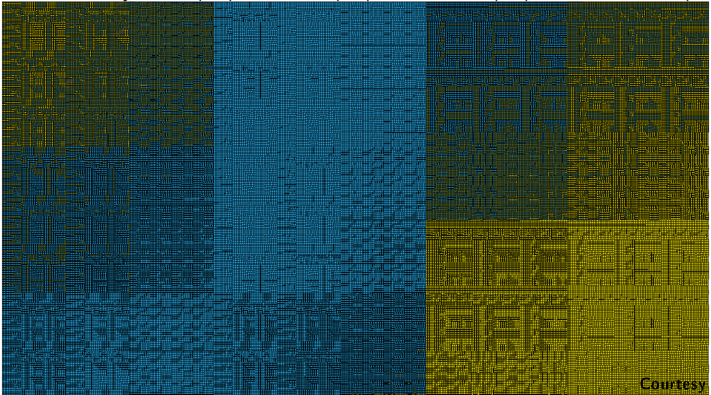
- Spatial and temporal trace aggregation
- Alternative visualization techniques
  - Squarified Treemap View
  - Hierarchical Graph View

Courtesy of Lucas Schnorr

Scenario 3 - Synthetic, Large Scale

- Synthetic trace with 100 thousand processes
- Two states, four-level hierarchy
- Visualization artifacts without spatial aggregation

A Hierarchy: Site (10) - Cluster(10) - Machine (10) - **Processor** (100)

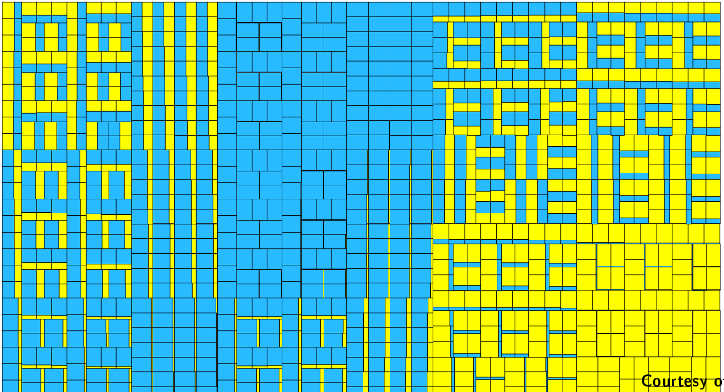


Courtesy of Lucas Schnorr

Scenario 3 - Synthetic, Large Scale

- Synthetic trace with 100 thousand processes
- Two states, four-level hierarchy
- Visualization artifacts without spatial aggregation

B Hierarchy: Site (10) - Cluster(10) - **Machine** (10) - Processor (100)

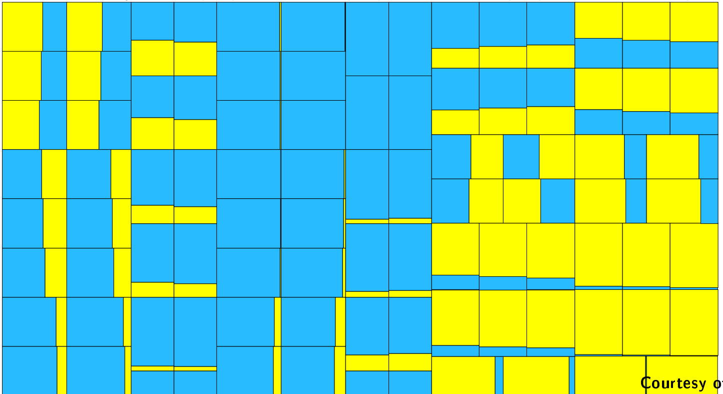


Courtesy of Lucas Schnorr

Scenario 3 - Synthetic, Large Scale

- Synthetic trace with 100 thousand processes
- Two states, four-level hierarchy
- Visualization artifacts without spatial aggregation

C Hierarchy: Site (10) - **Cluster**(10) - Machine (10) - Processor (100)

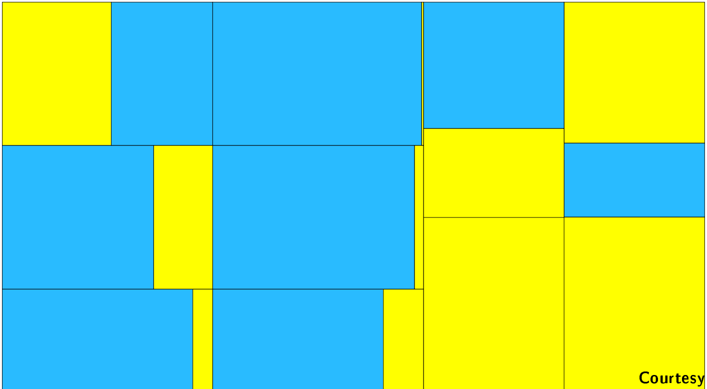


Courtesy of Lucas Schnorr

Scenario 3 - Synthetic, Large Scale

- Synthetic trace with 100 thousand processes
- Two states, four-level hierarchy
- Visualization artifacts without spatial aggregation

D Hierarchy: **Site** (10) - Cluster(10) - Machine (10) - Processor (100)

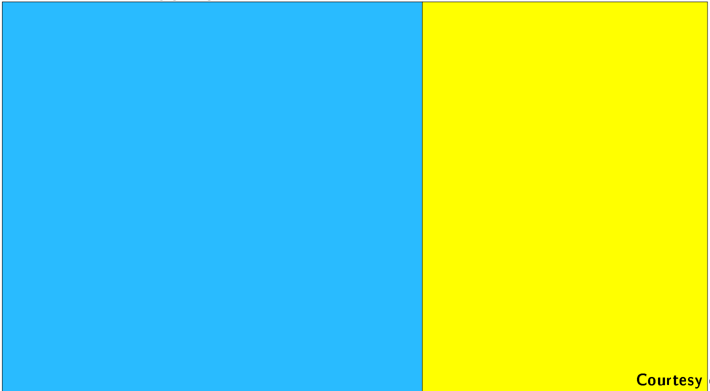


Courtesy of Lucas Schnorr

Scenario 3 - Synthetic, Large Scale

- Synthetic trace with 100 thousand processes
- Two states, four-level hierarchy
- Visualization artifacts without spatial aggregation

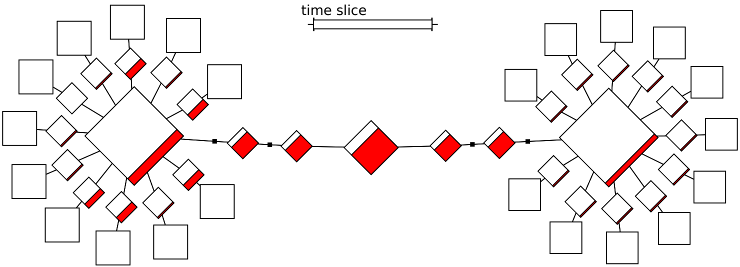
E Maximum Aggregation



Courtesy of Lucas Schnorr

Scenario 4 - NAS-DT Class A WH

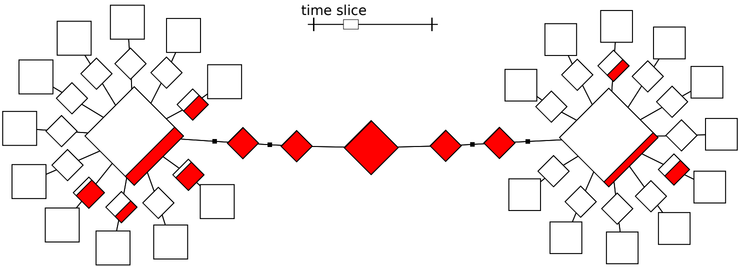
- NAS DT Class A White Hole algorithm
  $\rightarrow$ Traces from SMPI (Simulated MPI, part of SimGrid)
- Network topology – resource utilization by red filling
- Only temporal aggregation



Courtesy of Lucas Schnorr

Scenario 4 - NAS-DT Class A WH

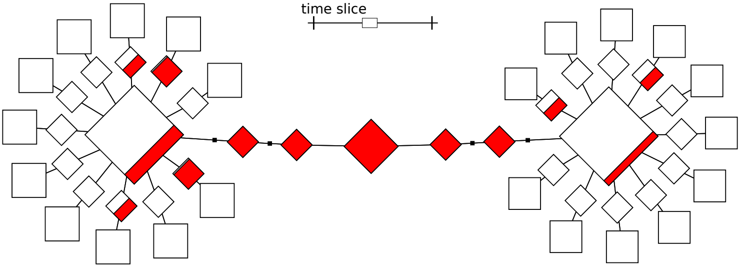
- NAS DT Class A White Hole algorithm
  → Traces from SMPI (Simulated MPI, part of SimGrid)
- Network topology – resource utilization by red filling
- Only temporal aggregation



Courtesy of Lucas Schnorr

Scenario 4 - NAS-DT Class A WH

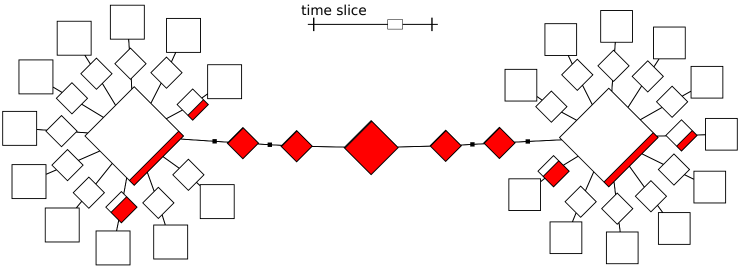
- NAS DT Class A White Hole algorithm
  → Traces from SMPI (Simulated MPI, part of SimGrid)
- Network topology – resource utilization by red filling
- Only temporal aggregation



Courtesy of Lucas Schnorr

Scenario 4 - NAS-DT Class A WH

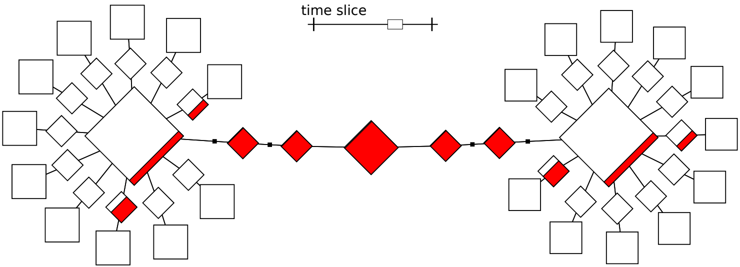
- NAS DT Class A White Hole algorithm
  → Traces from SMPI (Simulated MPI, part of SimGrid)
- Network topology – resource utilization by red filling
- Only temporal aggregation



Courtesy of Lucas Schnorr

Scenario 4 - NAS-DT Class A WH

- NAS DT Class A White Hole algorithm
  → Traces from SMPI (Simulated MPI, part of SimGrid)
- Network topology – resource utilization by red filling
- Only temporal aggregation

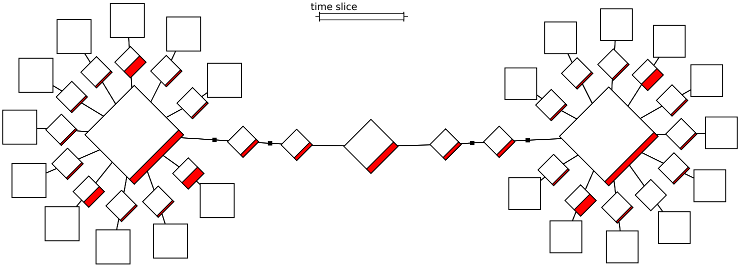


- Analysis: interconnection backbone is the bottleneck

Courtesy of Lucas Schnorr

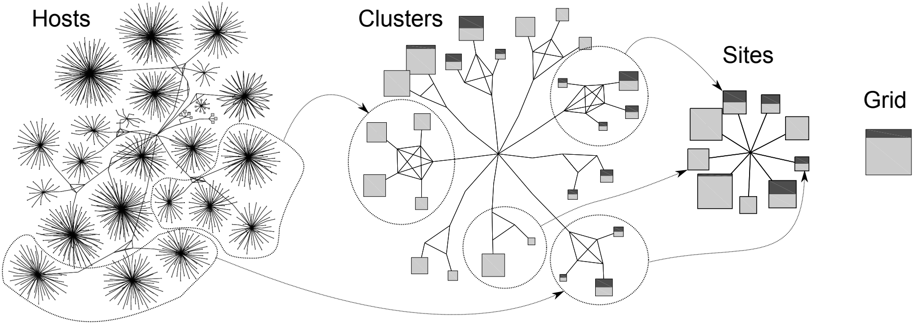
Scenario 4 - NAS-DT Class A WH (second try)

- Another deployment with a different mapping
  $\rightarrow$ by changing the order of machines in hostfile
- Explore communication locality



Courtesy of Lucas Schnorr

A visualization of G5K



Open-source tools

- Paje (Space/Time views, pie-charts), LGPL
  `http://paje.sourceforge.net`
  - Since 2000, GNUstep-based, written in Objective-C
  - Not only a monolithic visualization tool
    - Component-based, graph of components
    - Framework for developing other tools
    - Paje Protocol
  - 30K SLOC, hard to maintain, hard to install GNUstep

- Triva (Treemaps, Hierarchical graph), LGPL
  `http://triva.gforge.inria.fr`
  - Since 2007, GNUstep and Paje-based, also in Obj-C
    - Follows the Paje protocol
  - GNUstep runtime poses scalability problems

Courtesy of Lucas Schnorr

Current line of research

Use notions of Entropy to define meaningul aggegations and to guide aggregation.

- Robin Lamarche-Perrin (MAGMA/MESCAL): *Building Meaningful Macroscopic Descriptions of Large-scale Complex Systems*, October 2013.
- Damien Dosimont (MOAIS/MESCAL/NANOSIM – SocTrace)