High-performance **A**pplication and **C**omputers **S**tudying **PE**rformance and **C**orrectness **I**n **S**imulation

Paris, May 30th 2018

Meeting Schedule

10:00-13:00

- General presentation of HAC SPECIS
- Administrative Information
  - Positions, dissemination, collaborations, publications, ...
- A few success stories

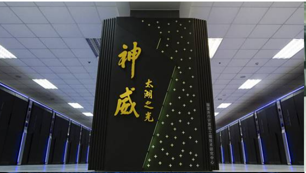
13:00 Lunch at "Le Repère"

14:30-17:00

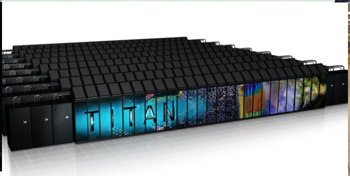
- 3 scientific focuses
- Perspectives & Discussions

Outline

Modern Supercomputers





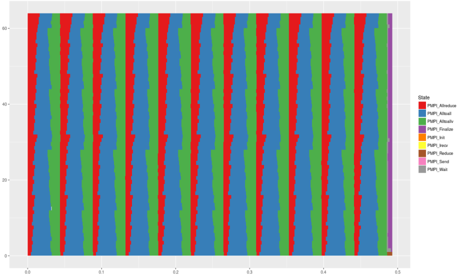


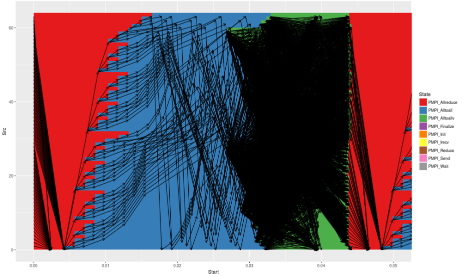
TaihuLight	40 960×260 (RISC)	Fat-Tree	15MW
Tianhe-2	32,000×12 (Xeon) + 48,000 Xeon Phi	Fat-Tree	18MW
Piz Daint	5,272×12 (Xeon) + 5,272 P100	DragonFly	2MW
Gyoukou	1,248×16 (Xeon D) + 9,984 PEZY-SC2 (1,984)	???	1MW
Titan	6,274×16 (AMD) + 18,688 K20	3D-Torus	8MW
Sequoia	98,304×12 (Power A2)	5D-Torus	8MW

- Up to 20,000,000 cores, accelerators, a complex high speed interconnect
- Co-design: Which technology for which application ? Energy/power management ?

Programming Supercomputers

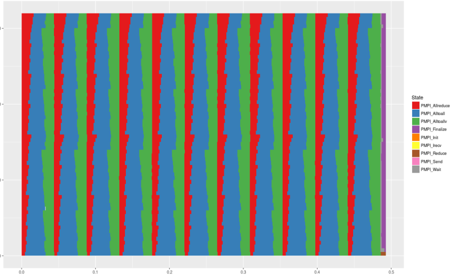
MPI (1994): $\rightsquigarrow$ regular algorithmic patterns

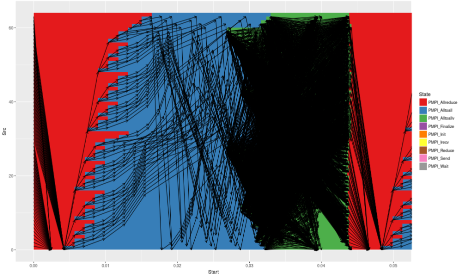




Programming Supercomputers

MPI (1994): $\rightsquigarrow$ regular algorithmic patterns





But many other APIs to exploit cores, GPUS, KNC, KNL, FPGAs, . . . :

OpenMP | CUDA | OpenCL | Cilk | Charm++ | KAAPI | StarPU | QUARK | ParSEC | OmpSs | ...

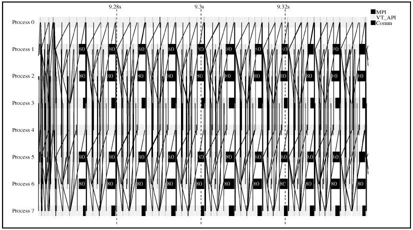
Exploitation is a programming (coding + algorithm) nightmare

Handling Supercomputers at the Application Level

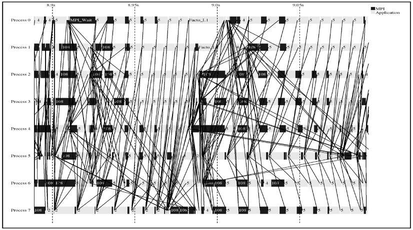
Larger and larger scale hybrid machines are a pain for application developers

- Programming models do not mix well
- Possible programming approaches

"Rigid, hand tuned" — "Task-based" and Dynamic



SuperLU



MUMPS

Analysis and Comparison of Two Distributed Memory Sparse Solvers
Amestoy, Duff, L'excellent, Li. ACM Trans. on Math. Software, Vol. 27, No. 4, 2001.

$\rightsquigarrow$ Applications are more and more adaptive

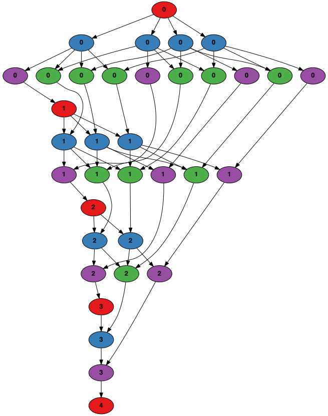
Toward Exascale

Modern Application Structure

1. Task-based Runtime (e.g., StarPU)
2. Task implementation Auto-Tuning

Typical Domains

- Dense linear solvers (e.g., Chameleon)
- Sparse linear solvers (e.g. `qr_mumps`), Low-rank
- Fast Multipole Methods
- Seismic application
- M.L. for Climate/Weather prediction



$\rightsquigarrow$ adaptive applications, ~~synchronizations~~, complex optimizations/scheduling

Performance ??? Correctness ???

Simulation & Model Checking

Performance Evaluation challenges

MPI/StarPU Simulation: what for ?

1. Helping application/runtime developers
   - Non-intrusive tracing, repeatable execution with classical debugging tools
   - Save computing resources (runs on your laptop if possible)
   - Provide a sound baseline for performance non regression testing
2. Helping application end-users
   - How much resources should I ask for? (scaling)
   - Configure MPI collective operations (tuning)
   - Provide baseline (did something go wrong ?)
3. Capacity planning
   - Energy consumption: can we save on components ?
   - Hardware upgrade: what-if the network was this ?

Could we have a tool that allows for this on a daily basis?

Formal verification challenges

MPI/StarPU Model Checking: what for ?

- Checking dynamic properties (deadlock, progression) of real HPC applications and runtimes
  - Heisenbugs are common
  - HPC applications are dirty but also have specific rigid structures
  - Adaptative applications require to explore all possible executions
- Quantitative properties (e.g., swapping)
  - Exact computation is painful
  - Estimate probability of particular events

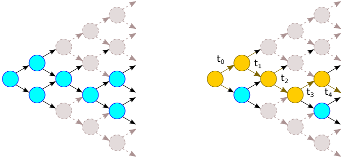
Could we have a tool that allows for this on a daily basis?

The need for specific tools/language is a showstopper for adoption in the HPC community $\rightsquigarrow$ Many almost unexplored problems on real apps

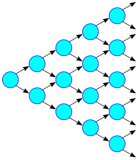
Global meetings help us discriminating short term and long term challenges

Simulation vs. Model Checking

- Simulation explores one possible execution of the program according to the features/limitations of the platform
- Model checking explores all possible executions of the program



State space with simulation



State space with model checking

- Simulation and model checking are complementary:
  - Simulation for performance evaluation
  - Model Checking for the verification of execution properties
- Both run automatically

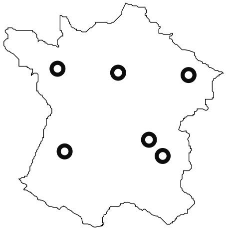
HAC Specis

HAC SPECIS Goal

- *Bridge the gap between the HPC*, formal verification† and performance evaluation§ communities*
- *Use SimGrid as an prototyping/integration platform*

Partners

- Rhône Alpes: AVALON$^{\star\ \S}$ POLARIS$^{\star\ \S}$ (+ CEA*)
- Bretagne Atlantique: MYRIADS$^{\star\ \dagger}$, SUMO†
- Sud Ouest: HIEPACS*, STORM*
- Île de France: MEXICO†
- Grand Est: VERIDIS†



A few dates

- Initial version in Jan 2014
- Final version in Dec 2016 (accepted April 2016)
- Kickoff June 2016 @ Rennes

Why Simgrid ?

SimGrid: A 18 years old open-source project. Collaboration between France (INRIA, CNRS, Univ. Lyon, Nancy, Grenoble, ...), USA (UCSD, U. Hawaii), UK, Austria (Vienna)...

http://simgrid.gforge.inria.fr

- 500 cite, 300 use, 60 extend
- Initially focused on Grid settings, we argue that the same tool/techniques can be used for P2P, HPC and cloud
- Goals: A usable tool with predictive capability
- SimGrid offers model checking capabilities since a few years

SimGrid 4 HPC

SMPI (2008-...) BigDFT, Ondes3D, SpecFEM3D, ...
StarPU-SimGrid (2013-...) Dense solvers, early work on `qr_mumps`

A Flourishing state of the Art

Many simulation projects for HPC

- Dimemas (BSC, probably one of the earliest)
- PSINS (SDSC, used to rely on Dimemas)
- BigSim (UIUC): BigNetSim or BigFastSim
- LogGopSim (UIUC/ETHZ)
- SST (Sandia Nat. Lab.)
- XSim (Oak Ridge Nat. Lab.)
- CODES (Argonne Nat. Lab.)

Verification for HPC

- CIVL (Delaware)
- MUST (Aachen)
- ISP and DAMPI (Utah)

Very few allow to work with real applications directly

Outline

- General Presentation
  - Context: Modern HPC
  - HAC SPECIS and SimGrid
- Administrative Facts
- Success Stories
  - Capacity planning with SMPI (Arnaud)
  - StarPU-SimGrid (Samuel)
  - Exact and Statistical Model Checking in Simgrid (Stephan)
  - Improving State Equality with Static Analysis (Emmanuelle)
  - Collateral Projects (Frédéric)
- Focus
  - Predicting Energy Consumption (Anne-Cécile & Arnaud)
  - StarPU-SimGrid (Samuel & Emmanuel)
  - Formal Aspects in HAC SPECIS (Martin & Thierry)
- Perspectives

Publications and Dissemination

Co-publications

- CCPE15, ICPADS15, Cluster17, TPDS17, CCPE18
- SC17: Correctness, VPA

Dissemination

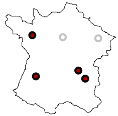
- SC16, SC17, ISC18 booths
- SIAMPP 2018: BoF on Simulation
- Intervention DGDT
- Maison simulation
- ...

Outline

- General Presentation
  - Context: Modern HPC
  - HAC SPECIS and SimGrid
- Administrative Facts
- Success Stories
  - Capacity planning with SMPI (Arnaud)
  - StarPU-SimGrid (Samuel)
  - Exact and Statistical Model Checking in Simgrid (Stephan)
  - Improving State Equality with Static Analysis (Emmanuelle)
  - Collateral Projects (Frédéric)
- Focus
  - Predicting Energy Consumption (Anne-Cécile & Arnaud)
  - StarPU-SimGrid (Samuel & Emmanuel)
  - Formal Aspects in HAC SPECIS (Martin & Thierry)
- Perspectives

HPL and the Top500

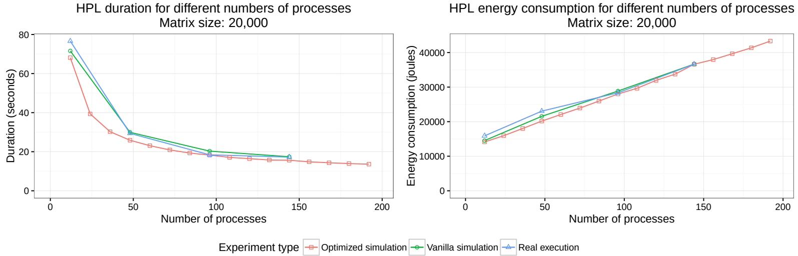
Context



- Real execution (qualification benchmark)
  - Matrix of rank 3,875,000: ≈ 120 Terabytes
  - 6,006 MPI processes for 2 hours: 500 CPU-days
- Simulation/Emulation with SMPI
  - 1 Xeon E5-2620 server (Nova, Grid'5000)
  - ≈ 47 hours and 16GB

Stampede, U.S.A., #20 with ≈ 5 Pflops
56 Gbit/s FDR InfiniBand Fat tree topology
6,400 × (8 cores + 1 Xeon Phi)

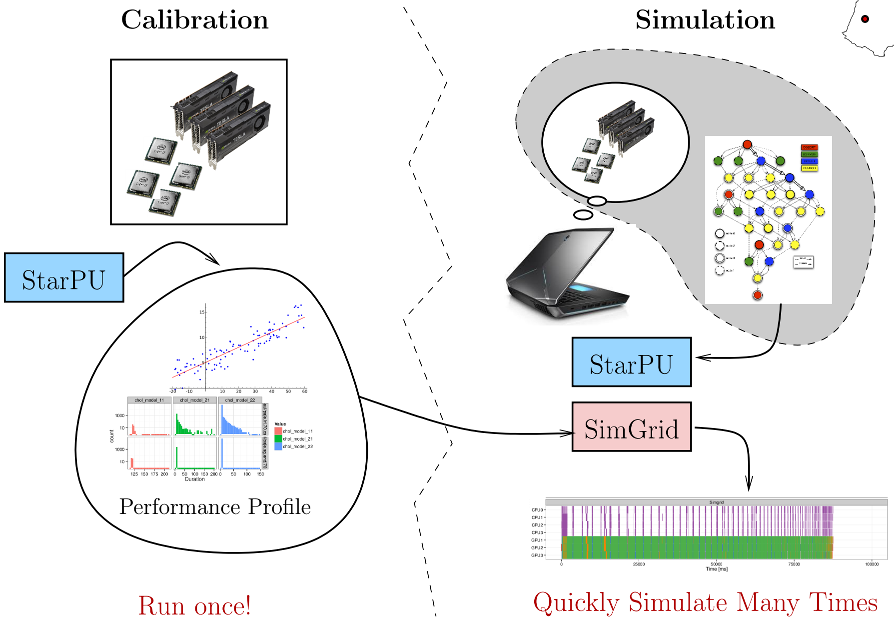
Accuracy (Evaluation on Taurus (Grid'5000))



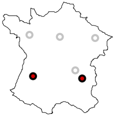
Mismatch with the Stampede qualification run (Intel HPL)

Perspective Capacity planning, Tune real applications, Co-Design, . . .

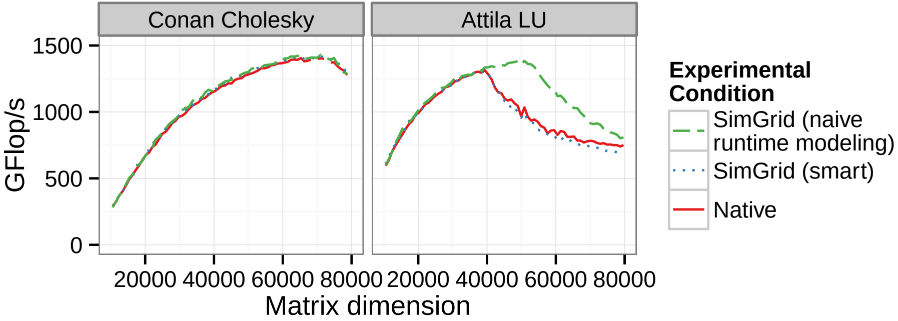
StarPU-Simgrid principle



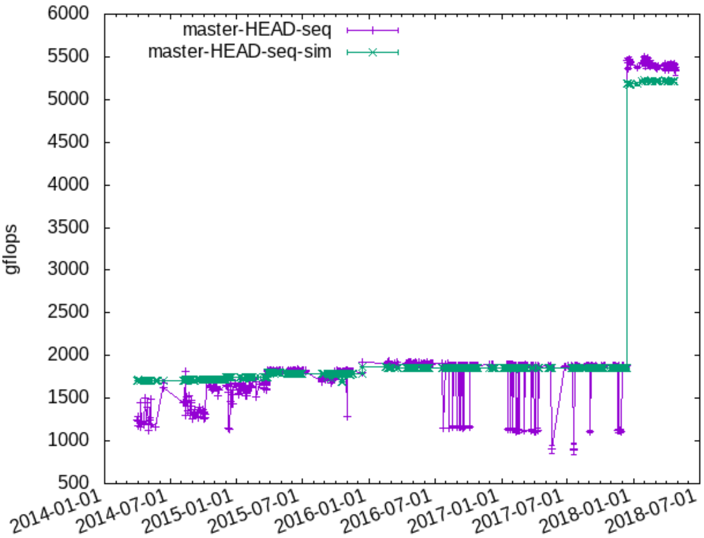
StarPU-Simgrid on dense linear algebra

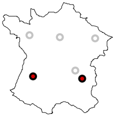


- Accurate simulated time results
- Already required a lot of care
- Extensively used for scheduling research



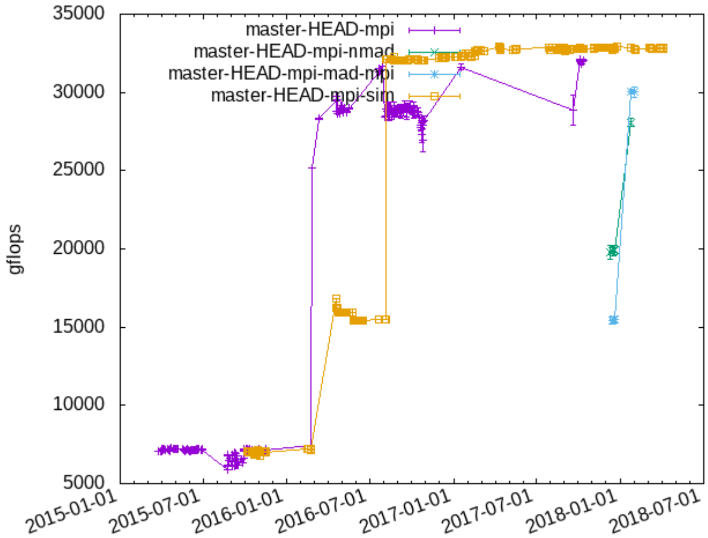
StarPU-Simgrid for CI





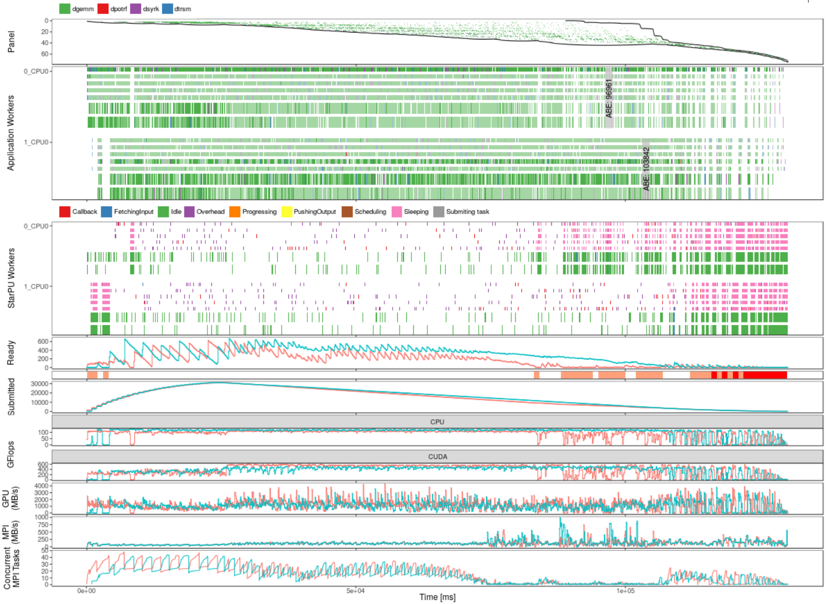
Execution on one node of sirocco (12 cores + 3 GPU K40M)

StarPU-Simgrid for CI with MPI

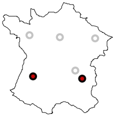


Execution on sirocco with 4 nodes (2×12 cores + 4 GPU K40M)

StarPU Visualization

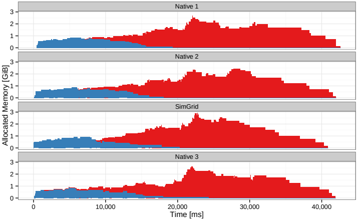


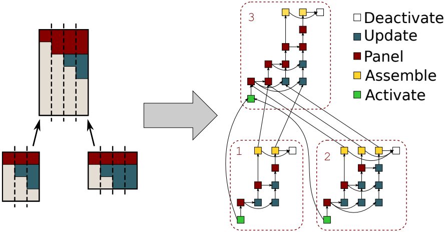
StarPU QR-Mumps

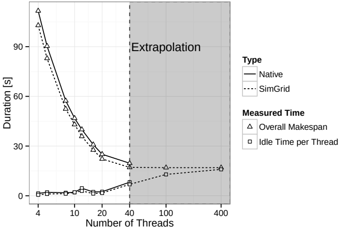


QR-MUMPS multi-frontal sparse factorization on top of StarPU

- Tree parallelism
- Node parallelism
- Variable matrix geometry
- Fully dynamic scheduling w. StarPU

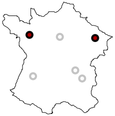






Perspective Tune app. and scheduler, capacity (memory) planning

Exact model checking

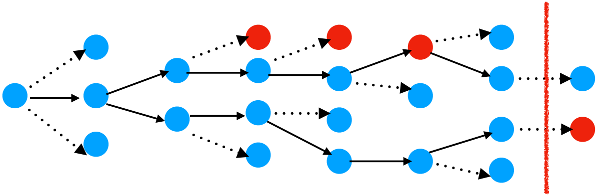


Explore reachable system states

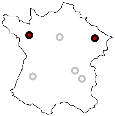
- check invariant properties in all states
- ensure liveness properties of infinite executions

Avoid computing and storing states when possible

- depth-bounded state space exploration
- save history of actions, recompute state when backtracking
- DPOR: avoid exploration of independent actions
- liveness checking: compress states, semantic equality checking



McSimGrid: Evaluation



McSimGrid is part of SimGrid distribution

Success stories

- identified reason of known bug in Chord implementation
- analyze conformance tests proposed for MPI implementations
- verification of send-determinism

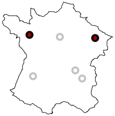
Drawbacks

- state space explosion restricts applicability to a handful of processes
- liveness properties are even harder to verify
- certain errors correspond to unrealistic border cases

Ongoing work

- use unfolding techniques to improve reduction (thèse The Anh Pham)
- static analysis for improving state equality detection (Emmanuelle Saillard)

Statistical Model Checking



Alternative to exhaustive verification of Boolean properties

- interval estimation: approximate a parameter value
- hypothesis testing: statistical evidence for acceptance or rejection
- quantify confidence in the result / probability of error

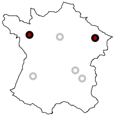
Perform Monte-Carlo simulation, apply statistical techniques

- no need to construct or store transition system
- applicable without prior knowledge about probability distributions
- simulation and analysis techniques are embarrassingly parallel

Combine performance evaluation and verification

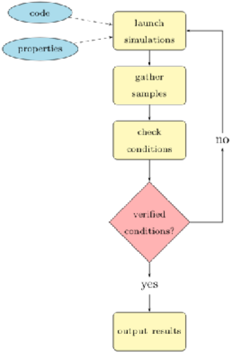
- estimate quantitative parameters, e.g. response time or memory consumption
- closely related to existing techniques within SimGrid

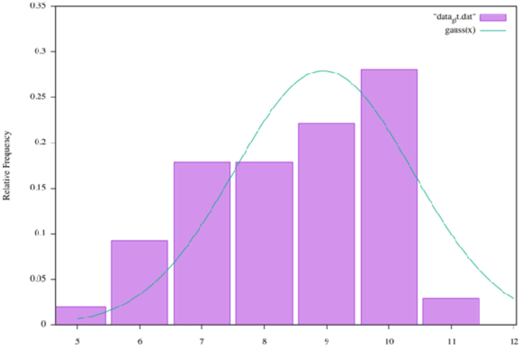
Preliminary Results



Prototypical implementation and application to Chord

- estimate average lookup time
- estimate percentage of failed lookups
- estimate network resilience (capacity to reconnect after failure)

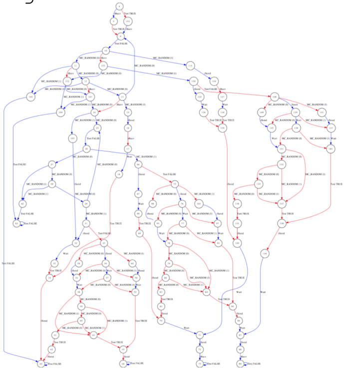




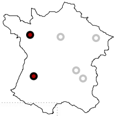
Mitigating the State Space Explosion

System-Level State Equality

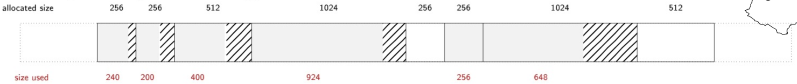
- Detect when a given state was previously explored
- Introspect the application state similarly to `gdb`
- Also with Memory Compaction



OS-level State Equality Detection

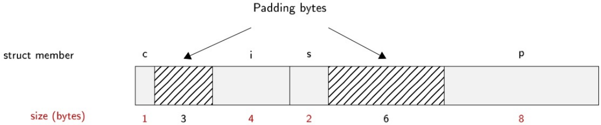


- Memory over-provisioning

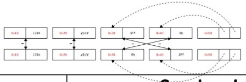


- Padding bytes: Data structure alignment

struct foo {
char c;
int i;
short s;
void *p;
}



- Irrelevant differences: system-level PID, fd, . . .
- Syntactic differences / semantic equalities:

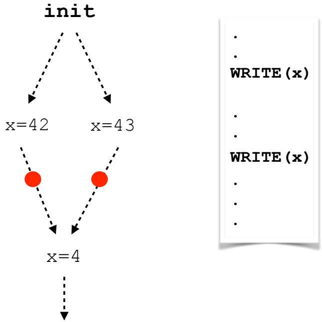


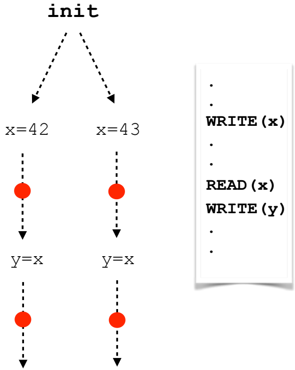
Solutions

Issue	Heap solution	Stack solution
Overprovisioning	`memset` 0 (customized `mmalloc`)	Stack pointer detection
Padding bytes	`memset` 0 (customized `mmalloc`)	DWARF + libunwind
Irrelevant differences	Ignore explicit areas	DWARF + libunwind + ignore
Syntactic differences	Heuristic for semantic comparison	N/A (sequential access)

Improve Dynamic State Equality Detection

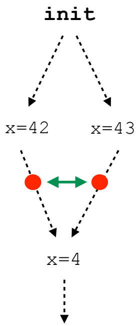
Reduce McSimGrid state space





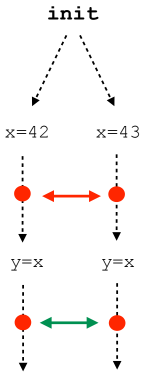
Improve Dynamic State Equality Detection

Reduce McSimGrid state space



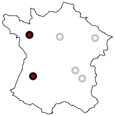

```
.
.
WRITE(x)

.
.
WRITE(x)
.
.
.
```

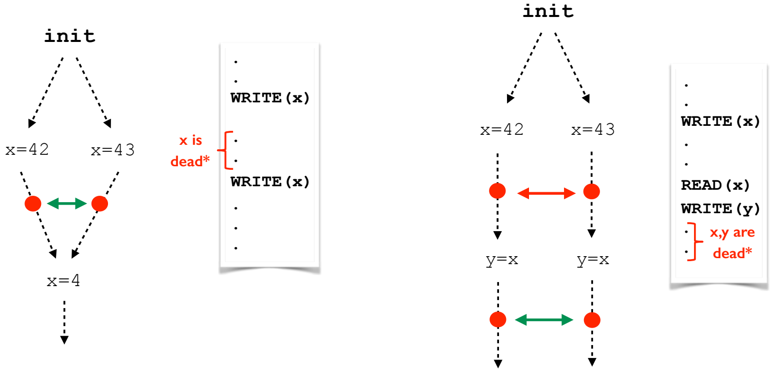



```
.
.
WRITE(x)
.
.
READ(x)
WRITE(y)
.
.
```

Improve Dynamic State Equality Detection

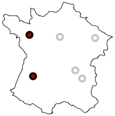


Reduce McSimGrid state space

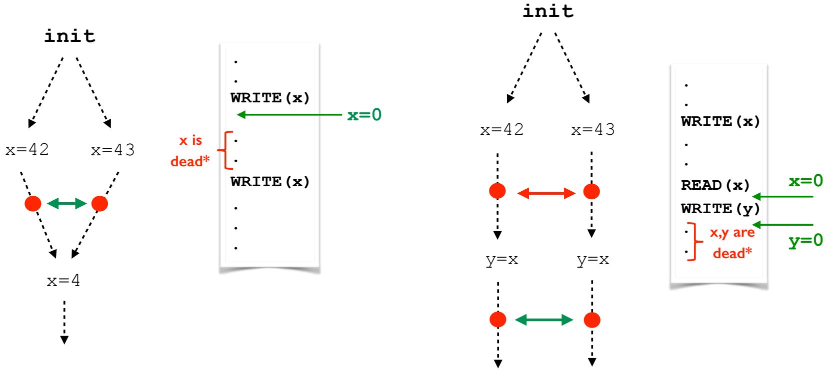


*redefined before it is used or never used in the future

Improve Dynamic State Equality Detection



Reduce McSimGrid state space



=> Set dead variables to 0

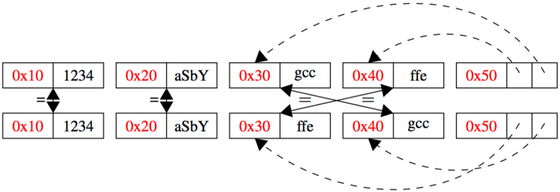
*redefined before it is used or never used in the future

6

Improve Dynamic State Equality Detection

Reduce McSimGrid state space

Give type information in the heap



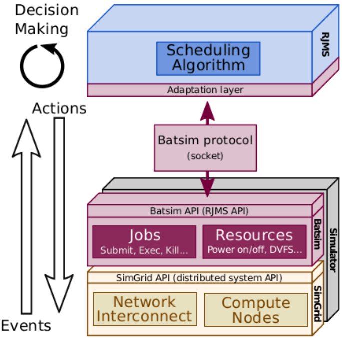
Two heaps syntactically different but semantically identical

=> Static Analysis in LLVM

BatSim

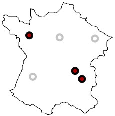
A Job and Resource Management System Simulator

- A key component in HPC systems
- Decouple the decision making from the simulation
- Uses SimGrid as a backend



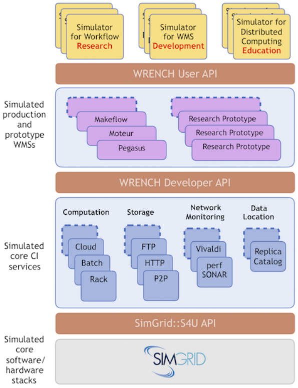
- Developed in the Datamove team (Grenoble)
- https://github.com/oar-team/batsim

Wrench



A Workflow Management System Simulation Workbench

- Objective
  - Provide high-level building blocks for developing custom simulators
- Targets:
  - Scientists: make quick and informed choices when executing workflows
  - Software developers: implement more efficient software infrastructures to support workflows
  - Researchers: Develop novel efficient algorithms
- Coupled with BatSim
- http://wrench-project.org
  - Collaboration with ISI/USC and UH Manoa
  - Funded by the NSF (grants number 1642369 and 1642335) and CNRS (PICS 7239)



Outline

- General Presentation
  - Context: Modern HPC
  - HAC SPECIS and SimGrid
- Administrative Facts
- Success Stories
  - Capacity planning with SMPI (Arnaud)
  - StarPU-SimGrid (Samuel)
  - Exact and Statistical Model Checking in Simgrid (Stephan)
  - Improving State Equality with Static Analysis (Emmanuelle)
  - Collateral Projects (Frédéric)
- Focus
  - Predicting Energy Consumption (Anne-Cécile & Arnaud)
  - StarPU-SimGrid (Samuel & Emmanuel)
  - Formal Aspects in HAC SPECIS (Martin & Thierry)
- Perspectives

Scientific Perspectives

Contribute to next-generation HPC tools & Gather strengths from 3 communities

Key scientific challenges:

Automatic performance modeling (building models, parameter learning)

- Platform (MPI)
- Computation kernels (StarPU)

OpenMP/SimGrid ($\rightsquigarrow$ PhD IPL 2018)

- Proxy Apps: 23/40 functional (mostly because of OpenMP)
- Capturing/modeling the impact of compiler/runtime, memory interferences

Improving DPOR ($\rightsquigarrow$ PhD IPL 2016)

- Good semantic formalization
- Optimality, merge with state equality, liveness, ...

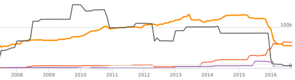
Applying "classical" safety MC techniques to HPC code ($\rightsquigarrow$ PostDoc IPL 2017)

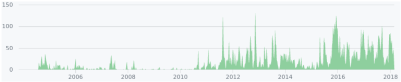
- Op. system challenges, improving state equality w. compiler
- Handle threads, checking applications and runtimes

Technical Considerations

Complex and Dynamic Code Base

- Only 100k sloc, but complex due to versatile efficiency + formal verification
- Implemented in C++/C (+ assembly); Bindings: Java, Lua and Fortran
- Active project: commits every day by $\approx$ 6 commiters, 4 releases a year
- Ongoing full rewrite in C++ along with *Release soon, Release often*

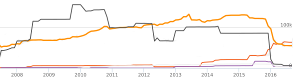


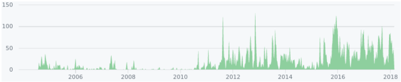


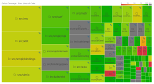
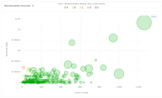
Technical Considerations

Complex and Dynamic Code Base

- Only 100k sloc, but complex due to versatile efficiency + formal verification
- Implemented in C++/C (+ assembly); Bindings: Java, Lua and Fortran
- Active project: commits every day by ≈ 6 commiters, 4 releases a year
- Ongoing full rewrite in C++ along with *Release soon, Release often*





Well Tested

- 740 integration tests, 10k units (coverage: 80%)
- Each commit: 22 configurations (4 OS, 3 compilers, 2 archs; 3 providers)
- Nightly: 2 dynamic + 2 static analyzers; StarPU, BigDFT and Proxy Apps

- We cultivate our garden: simplify to grow further

Building a Community

Communication and Animation

- SimGrid User Days: Welcome newcomers & Take feedback since 2010
- Scientific tutorials, Booth at SuperComputing & others; *Companies Courses*
- 500 cite 300 use 60 extend; 30 mails/month; 5 bugs/month; Stack Overflow

Preliminary Industrial Contacts

- CERN: currently testing the LHC DataGrid before production
- Intel/KAUST: internal project (est. at SC'17)
- Octo: dimensionning Ceph infrastructures for their clients (past attempt)
- Amazon/Nice: very preliminary contacts for dimensionning, service to clients
- My dream: make open-source IT infra (Samba, Ceph) testable with SimGrid
- Possible Income: subscription of 6-8 supporting institutions/companies

Toward Education

- Teach now the researchers and engineers of tomorrow to SimGrid
- Done: SMPI CourseWare, PeerSimGrid; Ongoing: Cloud, Wrench and more?