The SimGrid Project

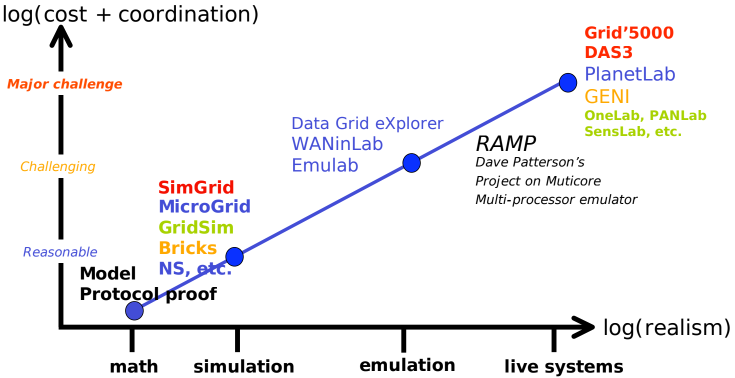
Algorille Team

Inria

October 11th, 2012

The Accuracy vs. Speed tradeoff

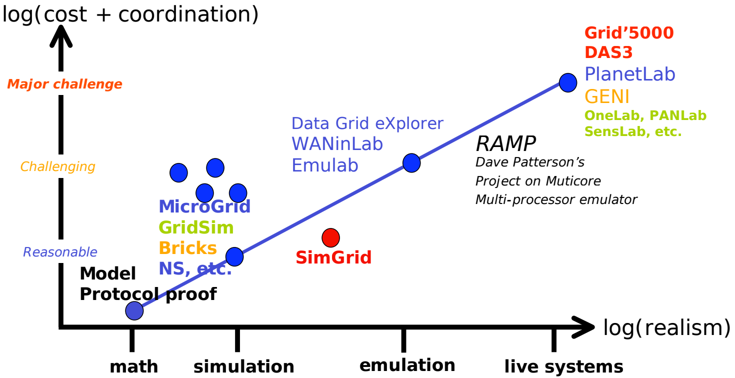
- Common Belief in 2008: Simulation as a toy methodology



Courtesy of Franck Cappello (Gri5000 keynote @ EGEE, Feb 2008)

The Accuracy vs. Speed tradeoff

- Common Belief in 2008: Simulation as a toy methodology
- Consensus in 2012: SimGrid as a scientific instrument (w/ Grid'5000)



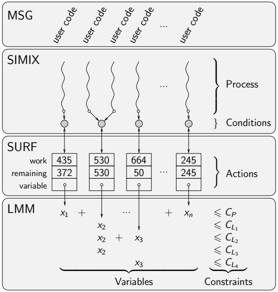
Purpose of this Talk

How did we turn **Simulation** into a
Reliable and Versatile Scientific Instrument
for Distributed Computing Research?

- A Performant et Versatile Simulation Kernel
  (high-performance simulation for computer science)
- Simulating Real MPI Applications
  (beyond prototypes)
- Toward a Coherent Workbench for Distributed Applications
  (when simulation is not enough)

Layered Infrastructure for a Versatile Tool

SimGrid: strictly layered and built bottom-up



SimGrid Functional Organization

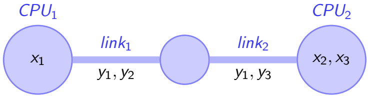
- Models: Actions get mapped onto resources
  Resource sharing and termination dates
- Activities: Processes interact and synchronize
- User interfaces: User-friendly syntaxic sugar

SimGrid user APIs

- SimDag: heuristics as DAG of (parallel) tasks
- MSG: heuristics as CSP (Java/Lua/Ruby bindings)
- SMPI: simulate MPI codes

Models: Resource Sharing between Actions

How to Model the Platform?



$$x_1 \leqslant Power\_CPU_1 \qquad (1a)$$

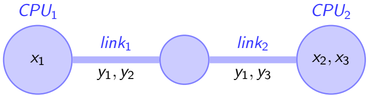
$$x_2 + x_3 \leqslant Power\_CPU_2 \qquad (1b)$$

$$y_1 + y_2 \leqslant Power\_link_1 \qquad (1c)$$

$$y_1 + y_3 \leqslant Power\_link_2 \qquad (1d)$$

Models: Resource Sharing between Actions

How to Model the Platform?



$$x_1 \leqslant Power\_CPU_1 \qquad (1a)$$
$$x_2 + x_3 \leqslant Power\_CPU_2 \qquad (1b)$$
$$y_1 + y_2 \leqslant Power\_link_1 \qquad (1c)$$
$$y_1 + y_3 \leqslant Power\_link_2 \qquad (1d)$$

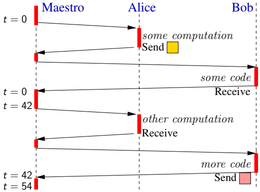
Production-grade Implementation

- Efficiency: Sparse structure; Cache oblivious; Lazy evaluation
- Realism: Several fairnesses can be expressed this way (or NS3 bindings)

Putting the Models in Use

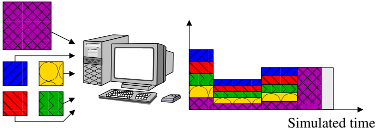
```
Alice
(some computation)
Send "toto" to Bob
(other computation)
Receive from Bob
```

```
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; update clock



Production-grade Implementation

- Scalability: Contextes instead of threads; Hierarchical networks
- Speed: Context switches in assembly; Futexes; Original parallelisation schema
- Other: Resource availability changes and failures; Dynamic Formal Verification

How big and how fast? (1/3 – Grid)

Size of platform description files

Community	Scenario	Size
P2P	2,500 peers with Vivaldi coordinates	294KB
VC	5120 volunteers	435KB + 90MB
Grid	Grid5000: 10 sites, 40 clusters, 1500 nodes	22KB
HPC	1 cluster of 262144 nodes	5KB
HPC	Hierarchy of 4096 clusters of 64 nodes	27KB
Cloud	3 small data centers + Vivaldi	10KB

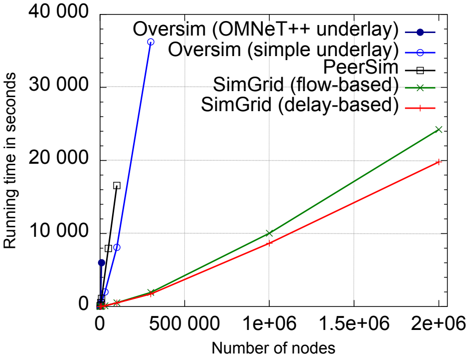
Speed of Grid Scenario

A master distributes 500, 000 fixed size jobs to 2, 000 workers (round robin)

	GridSim	SimGrid
Network model	delay-based model	flow model
Topology	none	Grid5000
Time	1h	14s
Memory	4.4GB	165MB

How big and how fast? (2/3 – P2P)

- Scenario: Initialize Chord, and simulate 1000 seconds of protocol
- Arbitrary Time Limit: 12 hours (kill simulation afterward)



Largest simulated scenario

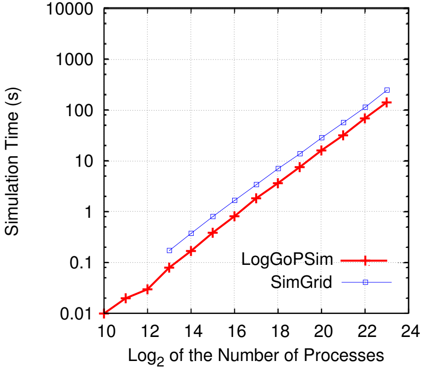
Simulator	size	time
OverSim (OMNeT++)	10k	1h40
OverSim (simple)	300k	10h
PeerSim	100k	4h36
SG (flow-based)	10k 300k 2M*	130s 32mn 6h23
SG (delay-based)	2M	5h30

∗ 36GB = 18kB/ process (16kB for the stack)

- SimGrid orders of magnitude more scalable than state-of-the-art P2P simulators
- Precise model incurs a ≈ 20% slowdown, but accuracy is not comparable

How big and how fast? (3/3 – HPC)

Simulating a binomial broadcast



Model:

- SimGrid: contention + cabinets hierarchy
- LogGOPSim: simple delay-based model

Results:

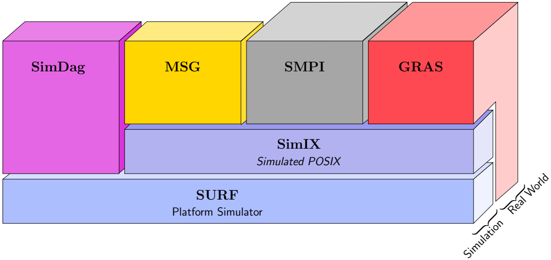
- SimGrid is roughly 75% slower
- SimGrid is about 20% more fat
  (15GB required for 2^{23} processors)

The genericity of SimGrid data structures comes at the cost of a slight overhead
BUT scalability does not necessarily comes at the price of realism

Conclusion

SimGrid is ready to ground your Research

- Versatile: Grid, P2P, HPC, Volunteer Computing, Clouds, ...
- Valid: Accuracy limits studied and pushed further for years
- Scalable: 3M chord nodes; 1000× faster than other (despite sound models)
- Usable: Tooling (generators, runner, vizu); Open-souce, Portable, ...



But a simulation kernel is not sufficient

- Users need love (Coming: Simulating MPI applications)
- Simulation is no universal solution (Coming: coherent workbench)

Single online simulation with SMPI

October 12th, 2012

On-line simulation in SMPI

- General Motivation: offer domain specific interfaces to SimGrid

On-line simulation in SMPI

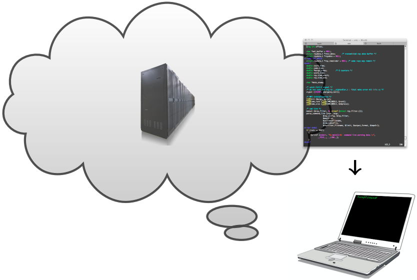
- General Motivation: offer domain specific interfaces to SimGrid
- SMPI: allows a user to simulate (possibly) unmodified MPI source code (C/Fortran)

On-line simulation in SMPI

- General Motivation: offer domain specific interfaces to SimGrid
- SMPI: allows a user to simulate (possibly) unmodified MPI source code (C/Fortran)
- Partial implementation of MPI on top of SimGrid

On-line simulation in SMPI

- General Motivation: offer domain specific interfaces to SimGrid
- SMPI: allows a user to simulate (possibly) unmodified MPI source code (C/Fortran)
- Partial implementation of MPI on top of SimGrid



On-line simulation in SMPI

- Computations: real execution on the host computer
  - CPU bursts are benched
  - Scale linearly CPU time according to power ratios

On-line simulation in SMPI

- Computations: real execution on the host computer
  - CPU bursts are benched
  - Scale linearly CPU time according to power ratios

- Communications: simulated
  - Network models are flow-based models (TCP)
  - Validity of these models for MPI applications

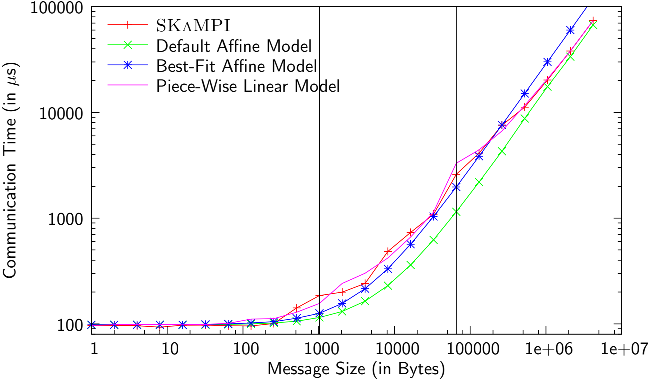
On-line simulation in SMPI

- Computations: real execution on the host computer
  - CPU bursts are benched
  - Scale linearly CPU time according to power ratios
- Communications: simulated
  - Network models are flow-based models (TCP)
  - Validity of these models for MPI applications
- Folding of the parallel program processes onto a single node
  - Serialization of computations
  - Single address space
  - Requires to reduce
    - Memory footprint (scalability)
    - Simulation time (speed)

Reworked Network Model

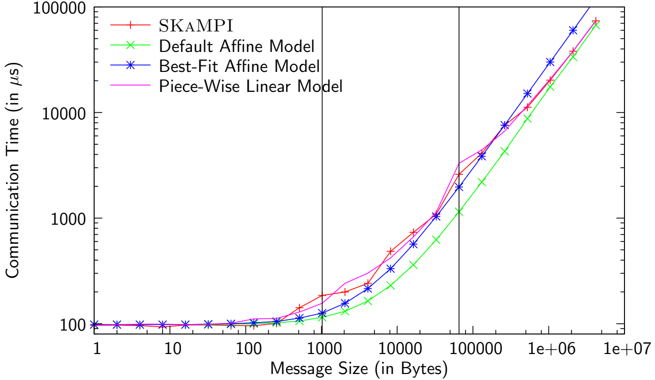
- Simple Model: $T(S) = L + \frac{S}{B}$
- Improved model: $T(S) = \alpha \cdot L + \frac{S}{min\left(\beta \cdot B, \frac{\gamma}{2 \cdot L}\right)}$
  - α accounts for TCP slow-start
  - β accounts for the overhead induced by TCP/IP headers (*e.g* 92%)
  - γ enables the modeling of the TCP window induced behavior
  - Model valid for $S \geq 100$ KiB, does not address a lot of message sizes found in MPI applications
- Need for a new, accurate network model when $S < 100$ KiB

Point-to-point Communication



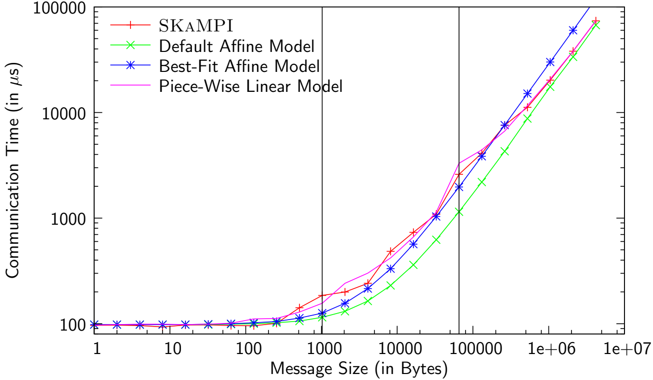
Experimental measurement using SKaMPI

Point-to-point Communication



Experimental measurement using SKaMPI
Default linear model, error: 32.1%
Ok with asymptotic message sizes,
but wrong for 1KiB-1MiB messages

Point-to-point Communication



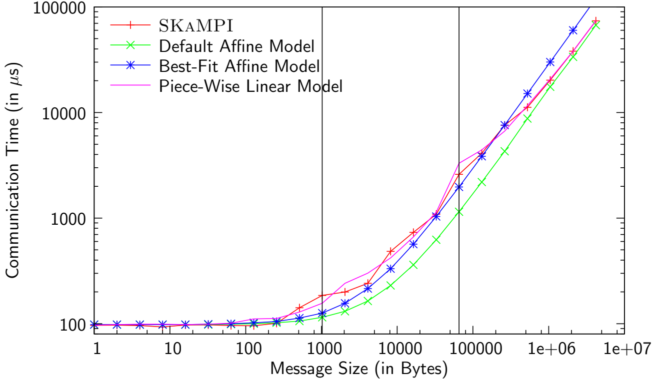
Experimental measurement using SKaMPI

Best-fitted linear model (α, β, γ), error: 18.5%

Better for a lot of sizes,

but cannot fit all real values

Point-to-point Communication



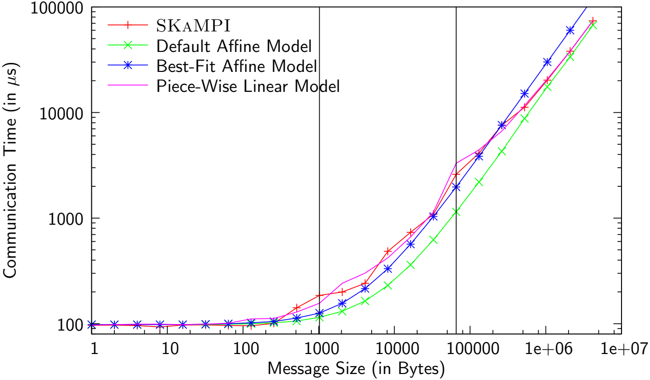
Experimental measurement using SKaMPI

Breakdown depending on message size

– packet size < MTU,

– eager/rendezvous switch limit

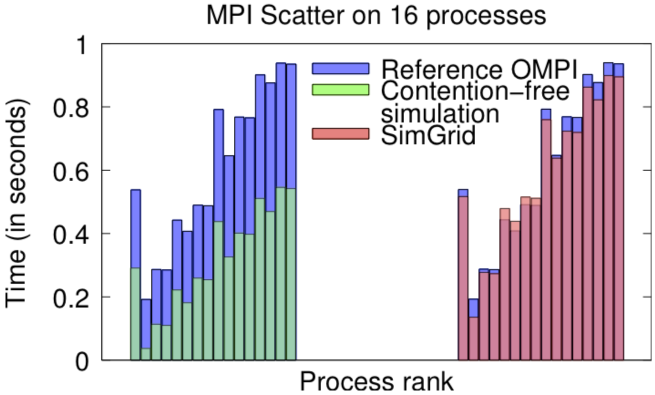
Point-to-point Communication



Experimental measurement using SKaMPI
New piece-wise linear model, error: 8.63%
Correctly adjust linear segments

Collectives and Contention

Scatter: 16-processes test



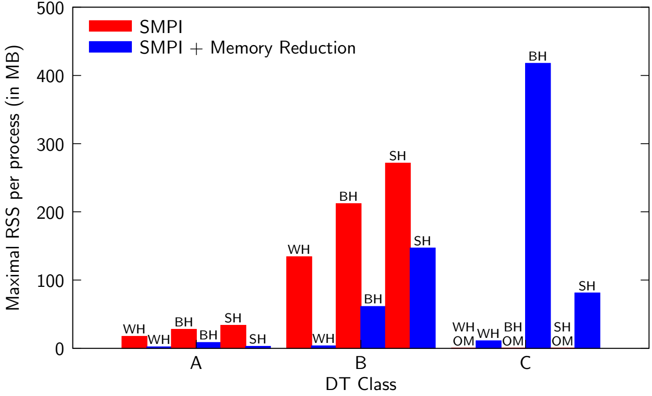
- Comparison SMPI/OpenMPI: error 5.3%
- Taking contention into account is important

Reducing the Memory Footprint

- Idea: Share arrays between processes
- Implemented as **optional** macros

```
double* data = (double*)SMPI_SHARED_MALLOC(...);
...
SMPI_SHARED_FREE
(data);
```

Reducing the Memory



- Average reduction by factor of 11.9 (maximum 40.5x)
- Class C can now be simulated

Reducing the Simulation Time

- Idea: Do not execute all the iterations
- Use sampling instead
  - LOCAL: each process executes a specified number of iterations
  - GLOBAL: a specified number of samples is produced by all processors
- Remaining iterations are replaced by average of measured values
- Implemented as **optional** macros

```
for(i = 0; i < n; i++) SMPI_SAMPLE_LOCAL( 0.75*n , 0.01 ) {   .
}
...
for(j = 0; j < k; j++) SMPI_SAMPLE_GLOBAL(0.5*k,0.01) {
   ...
}
```

Reducing the Simulation Time

- Idea: Do not execute all the iterations
- Use sampling instead
  - LOCAL: each process executes a specified number of iterations
  - GLOBAL: a specified number of samples is produced by all processors
- Remaining iterations are replaced by average of measured values
- Implemented as **optional** macros

```
for(i = 0; i < n; i++) SMPI_SAMPLE_LOCAL( 0.75*n , 0.01 ) {    .
}
...
for(j = 0; j < k; j++) SMPI_SAMPLE_GLOBAL(0.5*k,0.01) {
   ...
}
```

max part of iterations performed

threshold average variability

Wrap-up

- SMPI is a functional simulation tool
  - Open Source and freely available
  - Reproducible simulation of unmodified MPI application
  - On a single node
  - Main issues addressed:
    - scalability and speed through macros,
    - accuracy through extensions of the network model

Wrap-up

- SMPI is a functional simulation tool
  - Open Source and freely available
  - Reproducible simulation of unmodified MPI application
  - On a single node
  - Main issues addressed:
    - scalability and speed through macros,
    - accuracy through extensions of the network model
- However, microscopic behaviors are difficult to capture, e.g:
  - network communication jitters,
  - network catastrophes,
  - cache effects,
  - ...

Wrap-up

- SMPI is a functional simulation tool
  - Open Source and freely available
  - Reproducible simulation of unmodified MPI application
  - On a single node
  - Main issues addressed:
    - scalability and speed through macros,
    - accuracy through extensions of the network model
- However, microscopic behaviors are difficult to capture, e.g:
  - network communication jitters,
  - network catastrophes,
  - cache effects,
  - ...

And hence, simulation must be used in conjunction with other experimental approaches: emulation or experimentation in the real environment.

Experimentation methodologies in Algorille

Experimentation methodologies in Algorille

Simulator

- **M. Quinson** – core team,
  PI ANR SONGS 2012-2016
- **S. Genaud**, **J. Gossa**,
  L. Nussbaum also active

Experimentation methodologies in Algorille

Simulator

- **M. Quinson** – core team,
  PI ANR SONGS 2012-2016
- **S. Genaud**, **J. Gossa**,
  L. Nussbaum also active

Testbed

- **L. Nussbaum** – testbed design
  Proxy steering / tech. committees
- Team Focus on emulation and orchestration of experiments
- Engineering manpower (3 eng.)

Experimentation methodologies in Algorille

Simulator

- **M. Quinson** – core team, PI ANR SONGS 2012-2016
- **S. Genaud**, **J. Gossa**, **L. Nussbaum** also active

Testbed

- **L. Nussbaum** – testbed design *Proxy* steering / tech. committees
- Team Focus on emulation and orchestration of experiments
- Engineering manpower (3 eng.)

Key role in both projects

Experimentation methodologies in Algorille

Simulator

- **M. Quinson** – core team, PI ANR SONGS 2012-2016
- **S. Genaud**, **J. Gossa**, **L. Nussbaum** also active

Testbed

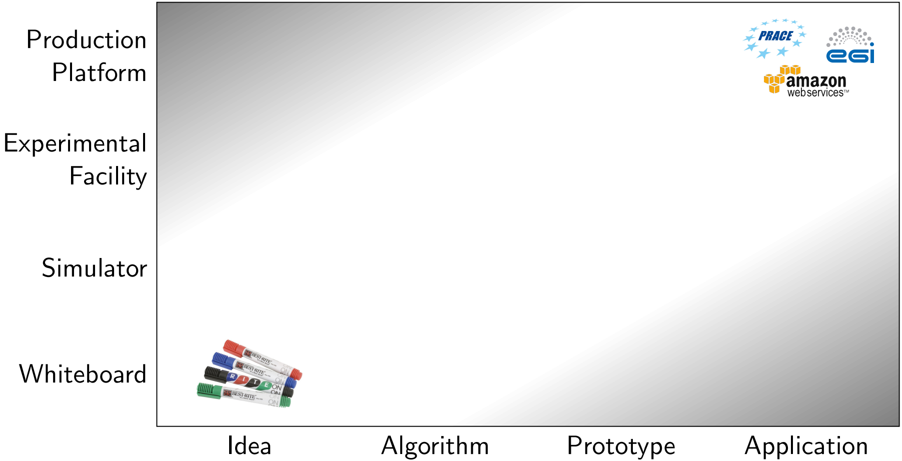
- **L. Nussbaum** – testbed design *Proxy* steering / tech. committees
- Team Focus on emulation and orchestration of experiments
- Engineering manpower (3 eng.)

Complementary solutions:

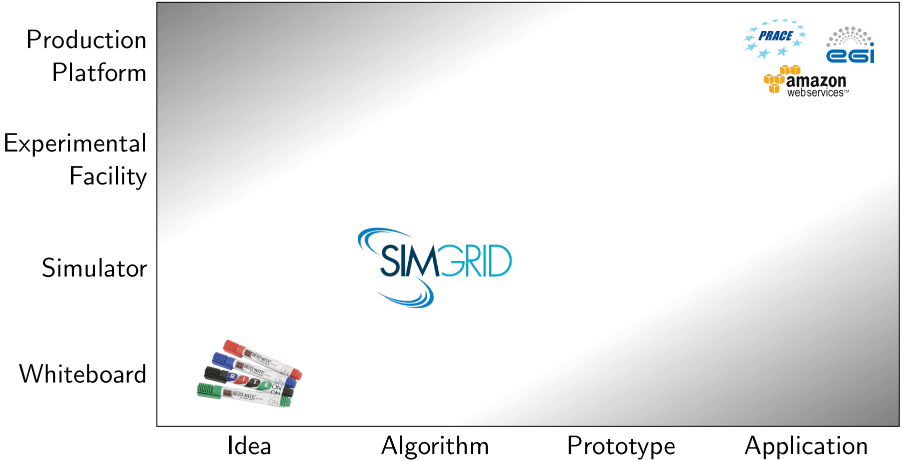
☺ Work on algorithms
☺ More scalable, easier

☺ Work on applications
☺ Perceived as more realistic

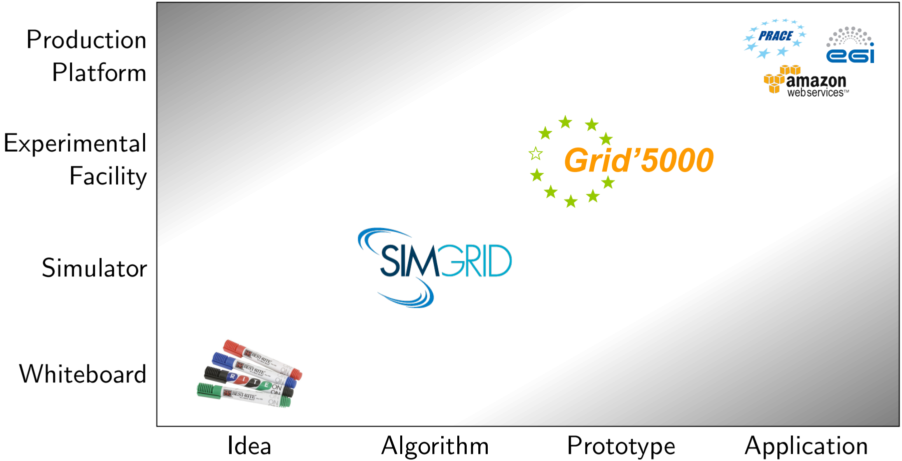
Leading users from ideas to applications



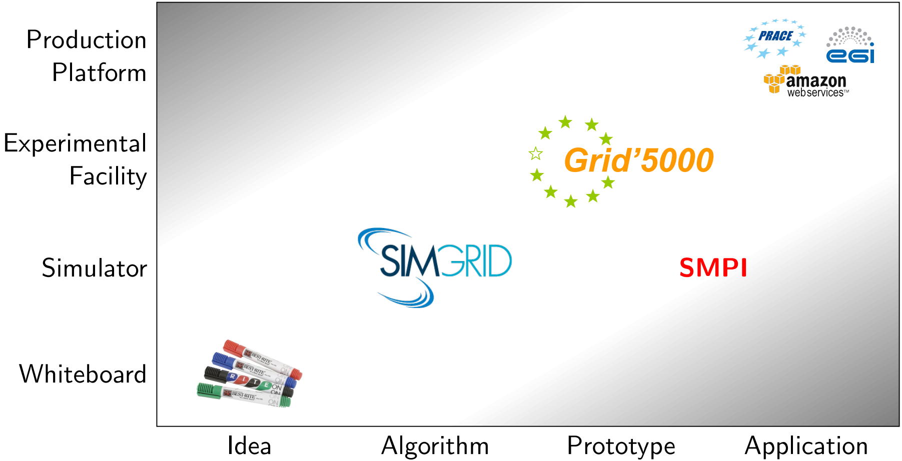
Leading users from ideas to applications



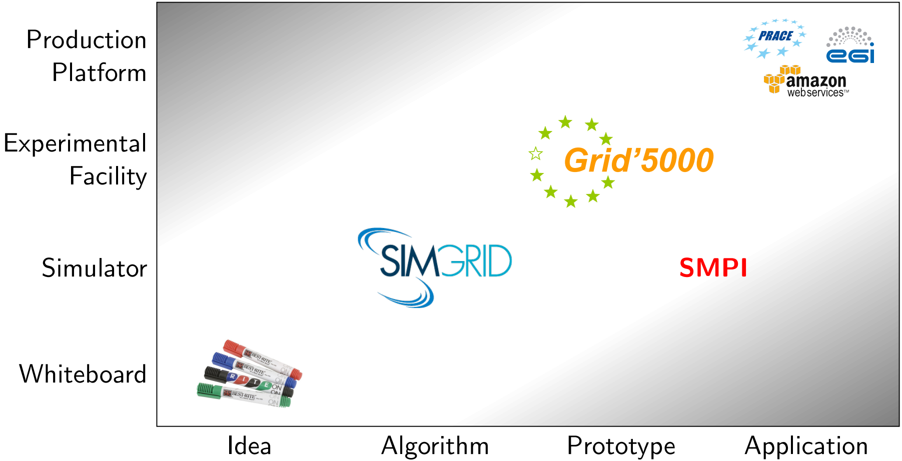
Leading users from ideas to applications



Leading users from ideas to applications



Leading users from ideas to applications



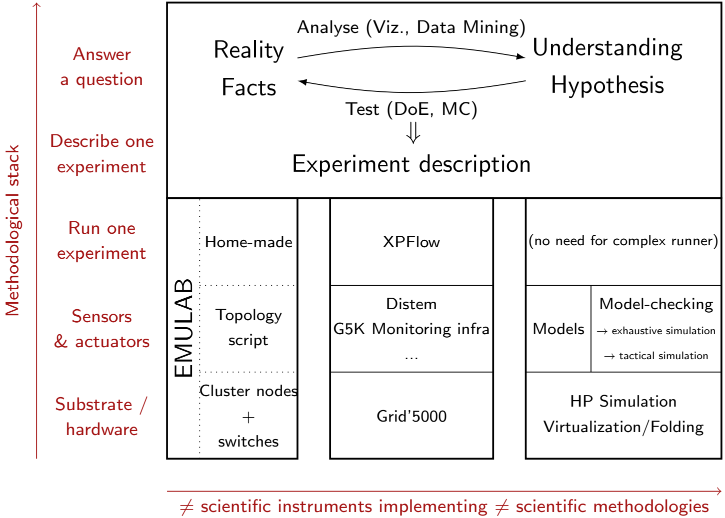
Goal: convergence of methodologies

Challenges and opportunities

- Share experimental methods and software
  - Infrastructure for Design of Experiment
  - Frameworks for data analysis and vizualisation

Challenges and opportunities

- Share experimental methods and software
  - Infrastructure for Design of Experiment
  - Frameworks for data analysis and vizualisation



Challenges and opportunities

- Share experimental methods and software
  - Infrastructure for Design of Experiment
  - Frameworks for data analysis and vizualisation
- Design better models and better testbeds using the common expertise
  e.g. network or power consumption modelling vs instrumentation
- Attack the same goals together, from both sides
  Reproducibility, trustworthiness, Open Science

We are in a unique position
to address those challenges