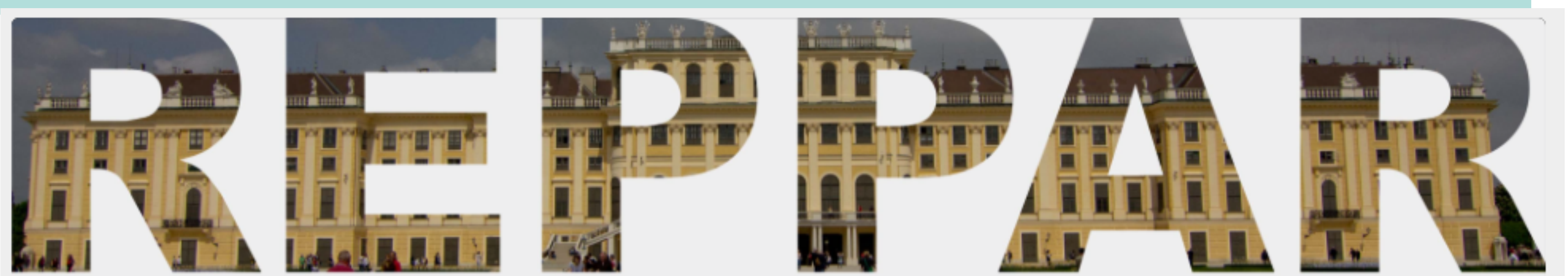
University of Basel

Reproducibility in Practice: Lessons Learned from Research and Teaching Experiments

Antonio Maffia, Helmar Burkhart and Danilo Guerrera,
Department of Mathematics and Computer Science
University of Basel, Switzerland
{antonio.maffia,helmar.burkhart,danilo.guerrera}@unibas.ch

REPPAR Workshop at Euro-Par 2015
August 25, 2015
Vienna, Austria



Prologue: Reproducibility - A Science Principle

"Non-reproducible single occurrences are of no significance to science". (Karl Popper The Logic of Scientific Discovery 1934/1959).

How Do Computational Disciplines Perform?

- **Algorithmic Engineering: J. of Experimental Algorithmics**
  www.jea.acm.org

  Authors are asked to make every effort to simplify the verification process*.*

Prologue: Reproducibility - A Science Principle

"Non-reproducible single occurrences are of no significance to science".
(Karl Popper The Logic of Scientific Discovery 1934/1959).

How Do Computational Disciplines Perform?

- Algorithmic Engineering: J. of Experimental Algorithmics www.jea.acm.org
- **Programming Languages & Methodologies: http://evaluate.inf.usi.ch**

  Canon (What is good science?); Artifacts (Software, Data): Evaluation and Certificate.

Prologue: Reproducibility - A Science Principle

"Non-reproducible single occurrences are of no significance to science".
(Karl Popper The Logic of Scientific Discovery 1934/1959).

How Do Computational Disciplines Perform?

- Algorithmic Engineering: J. of Experimental Algorithmics www.jea.acm.org
- Programming Languages & Methodologies: http://evaluate.inf.usi.ch
- **Artificial Intelligence: http://recomputation.org**

  Manifesto ("If we can compute your experiment now, anyone can recompute it 20 years from now"), Virtual Machine ("the only way", "Runtime performance is a secondary issue").

Prologue: Reproducibility - A Science Principle

"Non-reproducible single occurrences are of no significance to science". (Karl Popper The Logic of Scientific Discovery 1934/1959).

How Do Computational Disciplines Perform?

- Algorithmic Engineering: J. of Experimental Algorithmics www.jea.acm.org
- Programming Languages & Methodologies: http://evaluate.inf.usi.com
- Artificial Intelligence: http://recomputation.org
- **Computational Sciences: Tools Available.**

Workbench Approach: Taverna, Vistrails, Kepler, Knime
Version Control Approach: Sumatra, Madagascar
Virtualization Approach: CDE, Emulab

Prologue: Reproducibility - A Science Principle

"Non-reproducible single occurrences are of no significance to science".
(Karl Popper The Logic of Scientific Discovery 1934/1959).

How Do Computational Disciplines Perform?

- Algorithmic Engineering: J. of Experimental Algorithmics www.jea.acm.org
- Programming Languages & Methodologies: http://evaluate.inf.usi.com
- Artificial Intelligence: http://recomputation.org
- Computational Sciences: Tools Available.

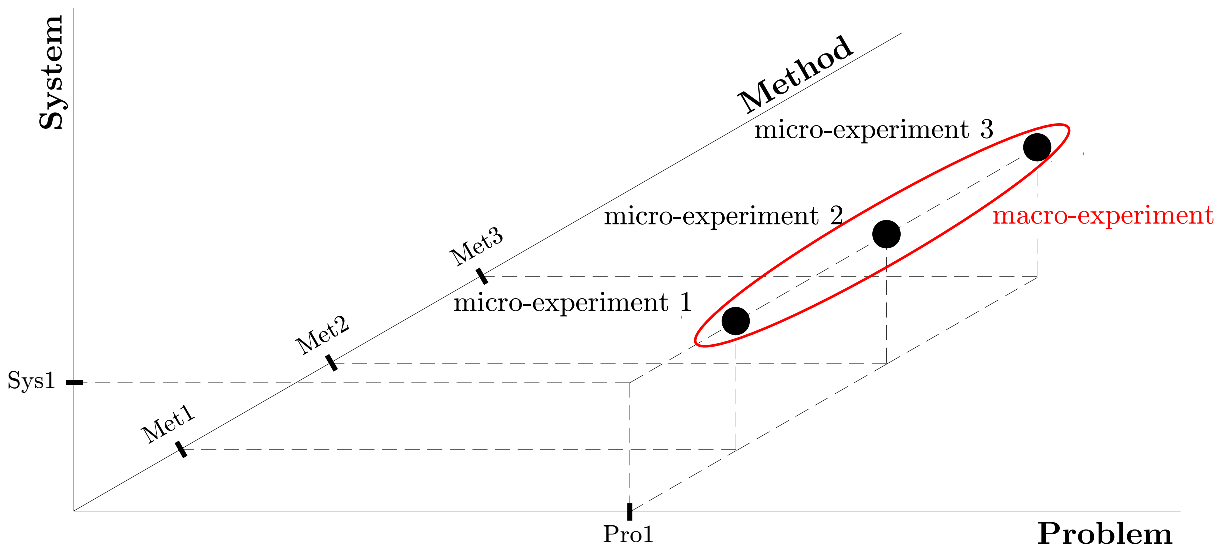
Good ad-hoc efforts, but none is addressing Performance (benchmarking)!

Outline for the Next 20 Minutes

- The Basel Taxonomy for Computational Experiments
- PROVA! Architecture of our Prototype System
- 4 Case Studies of Repeatability
- Demonstration of Replication
- Conclusions
- Future Work
- Discussion

The Basel Taxonomy[1] for Computational Experiments

[1] *D. Guerrera, H. Burkhart, and A. Maffia:* Reproducible Experiments in Parallel Computing: Concepts and Stencil Compiler Benchmark Study, Reppar Workshop 2014



Experiment definition

– **Problem**: specification of the problem including characteristic parameters.

– **Method**: description of the algorithmic approach used to tackle the problem.

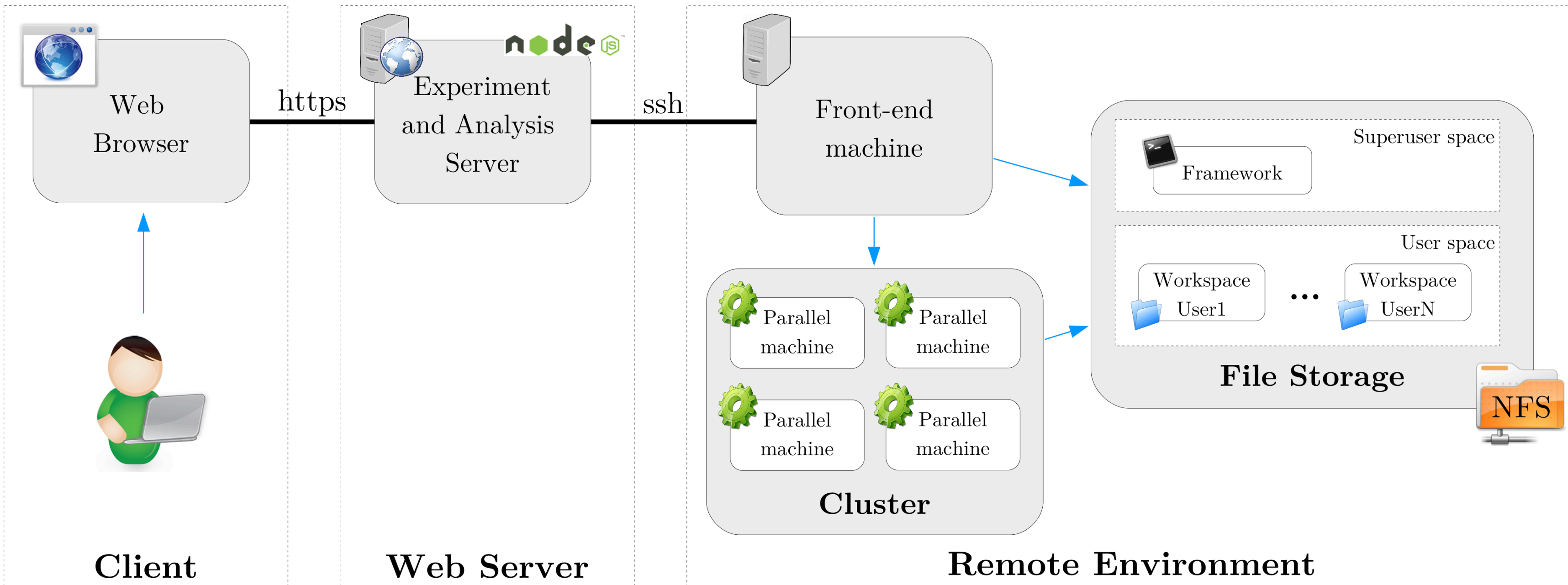
– **System**: representation of the compute environment (both hardware and software), on which an experiment is run.

Repeatability levels

– Re-run the same experiment or change parameters' values (***Replication***)
– Change the system (***Recomputation***)
– Change the method (***Reproduction***)

PROVA![1] – Architecture of our Prototype System

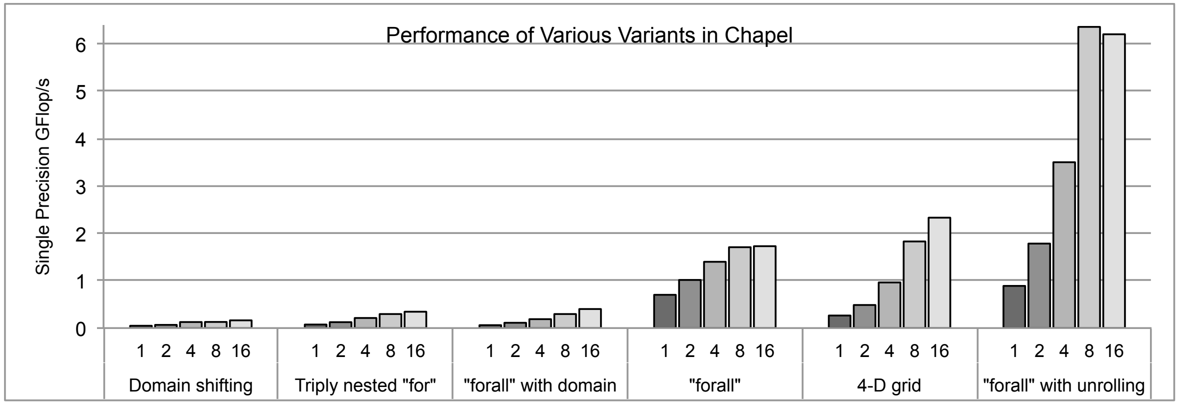
[1] from the Italian: try (the tool), prove (your results), and thus convince me.



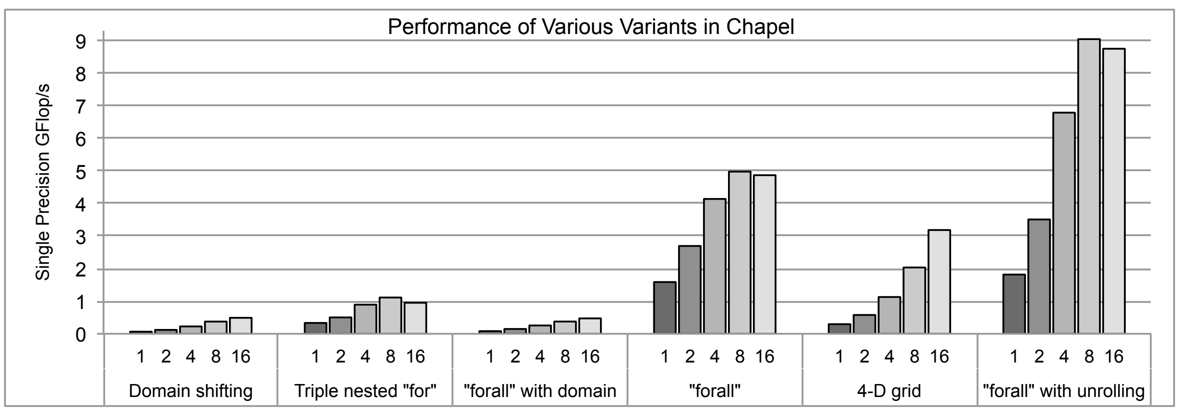
Case Study 1 – Research Result Review

	2012	2015
Experiment definition	**PGAS12 Conference Paper** *Original experiment*	**University of Basel (HPWC Group)** *Recomputation*
Problem	Calculate a 3-D wave equation of 200^3 elements (IEEE double precision arithmetic) in 100 timesteps.	**Unchanged**
System	**SW**: PGAS programming model using Chapel v1.4, GCC unknown **HW**: 1 node - CPU: Intel Core i7 4-Core (8 threads), 2.67 GHz, cache unknown - RAM: unknown - OS: unknown	**SW**: PGAS programming model using Chapel v1.4, GCC 4.7.2 **HW**: 1 node - CPU: 2x AMD Opteron 6274 16-Core, 2.2 GHz, 12 MiB L3 cache - RAM: 256 GiB - OS: Ubuntu 12.04.4, Kernel 3.8.0-38
Method	**1.** Use “domain translation” function **2.** Use 3 nested “for” loops **3.** Use “forall” statement (explicit indices) **4.** Use “forall” statement with “domain map” function **5.** Use a 4D grid swapping indices instead of grids after timestep **6.** Use “forall” unrolling 3 grid computation	**Unchanged**

Case Study 1 – Research Result Review



Original results graph. PGAS12 Conference Paper (2012)[1].



Recomputed results graph. University of Basel (HPWC Group).

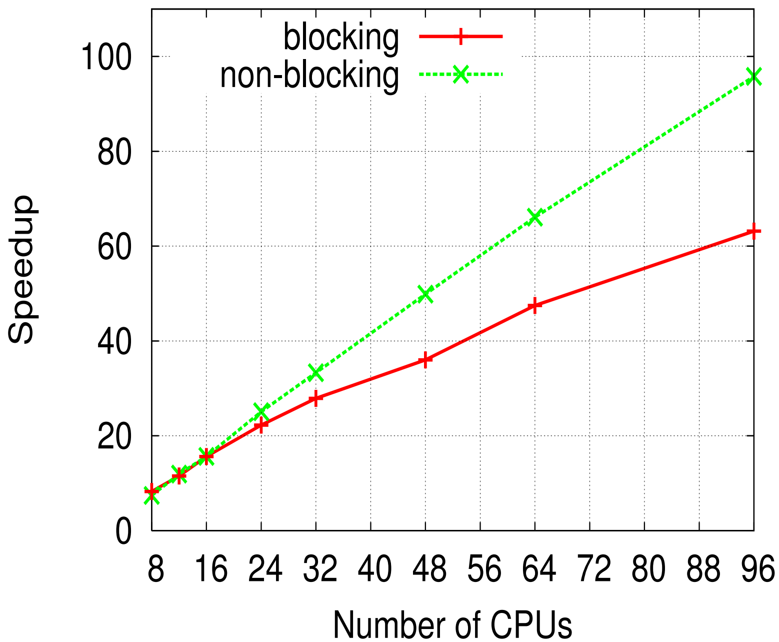
[1]*H. Burkhart, M. Sathe, M. Christen, M. Rietmann, and O. Schenk*: Run, Stencil, Run!—HPC Productivity Studies in the Classroom, Proc. 6th Conference on Partitioned Global Address Space Programming Models (PGAS12)

Case Study 2 – Research Result Review

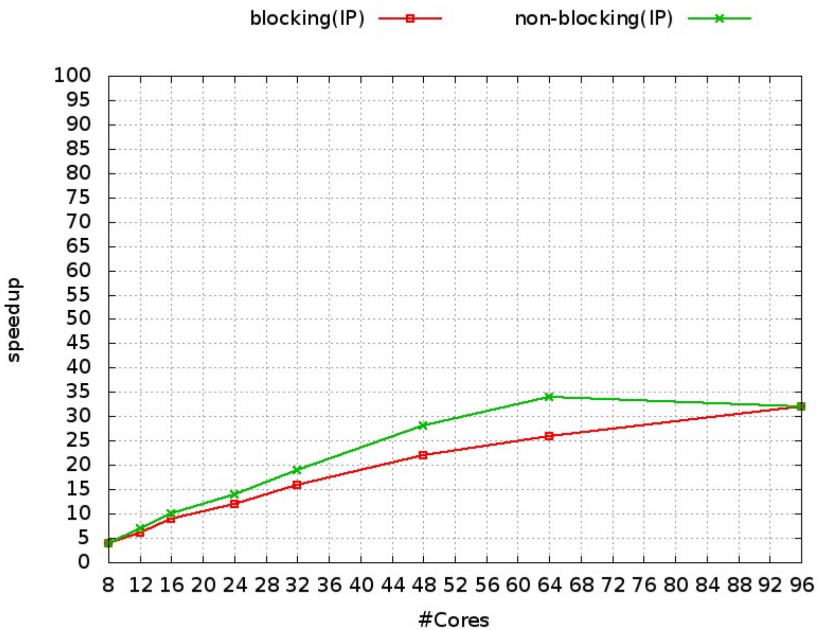
	2007	2014
Experiment definition	**PARCO Paper** *Original experiment*[1]	**University of Potsdam** *Recomputation*[2]
Problem	Calculate a 3-D Poisson equation of 800^3 elements (IEEE double precision arithmetic) with an error of 0.01.	**Unchanged**
System	**SW**: Distributed-memory computer with Message Passing Interface (MPICH2 1.0.2), GCC unknown **HW**: 128 nodes - CPU: dual Opteron 246, 2 GHz, cache unknown - RAM: unknown - OS: unknown	**SW**: Distributed-memory computer with Message Passing Interface (MPICH 3.1.2), GCC unknown **HW**: 28 homogeneous nodes (Dell R610) - CPU: 2x Intel Xeon E5520 (Nehalem) 4-Core (SMT deactivated), 2,26 GHz, 8 MiB L3 cache - RAM: 48 GiB - OS: Scientific Linux, Kernel 2.6.18
Method	**1.** Blocking collective communication **2.** Non-blocking collective communication using NBC library	**Unchanged**

[1]*Hoefler et al.*: Optimizing a Conjugate Gradient Solver with Non-Blocking Collective Operations
[2]*Schnor et al.*: Non-Blocking collectives. Seminar at University of Potsdam

Case Study 2 – Research Result Review



Original results graph.
PARCO Paper (2007)[1]



Recomputed results graph.
University of Potsdam (2014)[2]

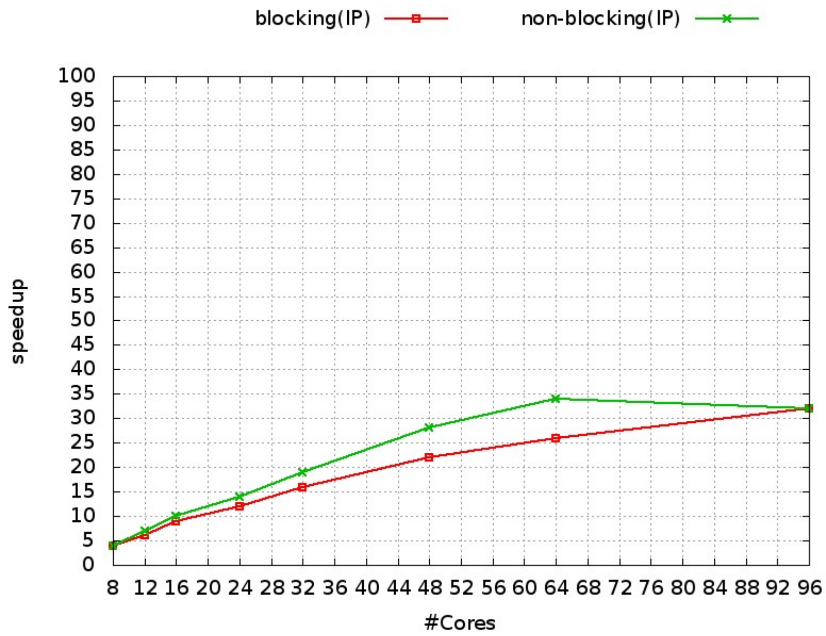
[1]*Hoefler et al.*: Optimizing a Conjugate Gradient Solver with Non-Blocking Collective Operations
[2]*Schnor et al.*: Non-Blocking collectives. Seminar at University of Potsdam

Case Study 2 – Research Result Review

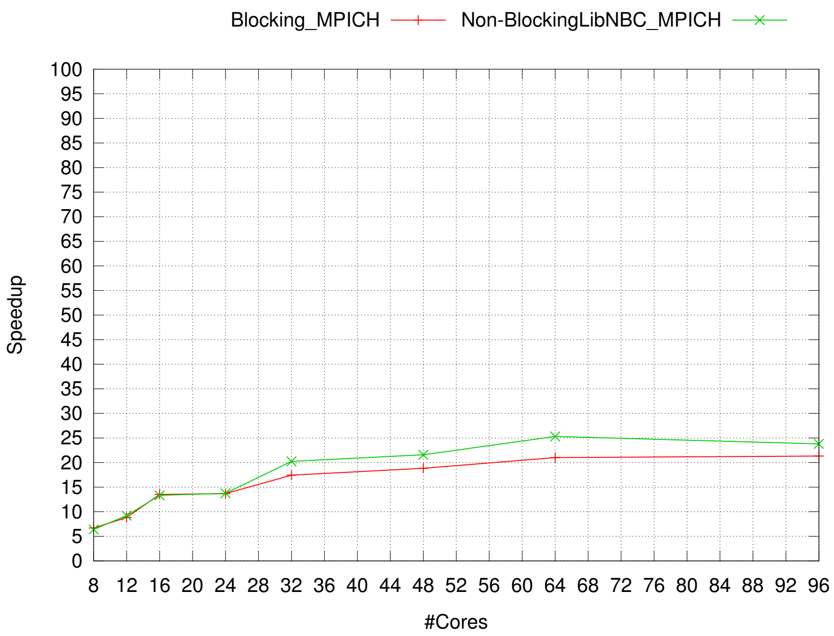
	2014	2015
Experiment definition	**University of Potsdam** *Original experiment*	**University of Basel (HPWC Group)** *Recomputation*
Problem	Calculate a 3-D Poisson equation of 800^3 elements (IEEE double precision arithmetic) with an error of 0.01.	**Unchanged**
System	**SW**: Distributed-memory computer with Message Passing Interface (MPICH 3.1.2), GCC unknown **HW**: 28 homogeneous nodes (Dell R610) - CPU: 2x Intel Xeon E5520 (Nehalem) 4-Core (SMT deactivated), 2,26 GHz, 8 MiB L3 cache - RAM: 48 GiB - OS: Scientific Linux, Kernel 2.6.18	**SW**: Distributed-memory computer with Message Passing Interface (MPICH 3.1.4), GCC 4.7.2. **HW**: 4 homogeneous nodes - CPU: 2x AMD Opteron 6274 16-Core, 2.2 GHz, 12 MiB L3 cache - RAM: 256 GiB - OS: Ubuntu 12.04.4, Kernel 3.8.0-38
Method	**1.** Blocking collective communication **2.** Non-blocking collective communication using NBC library	**Unchanged**

[1] *Schnor et al.*: Non-Blocking collectives. Seminar at University of Potsdam

Case Study 2 – Research Result Review



Original results graph.
University of Potsdam (2014)[1].



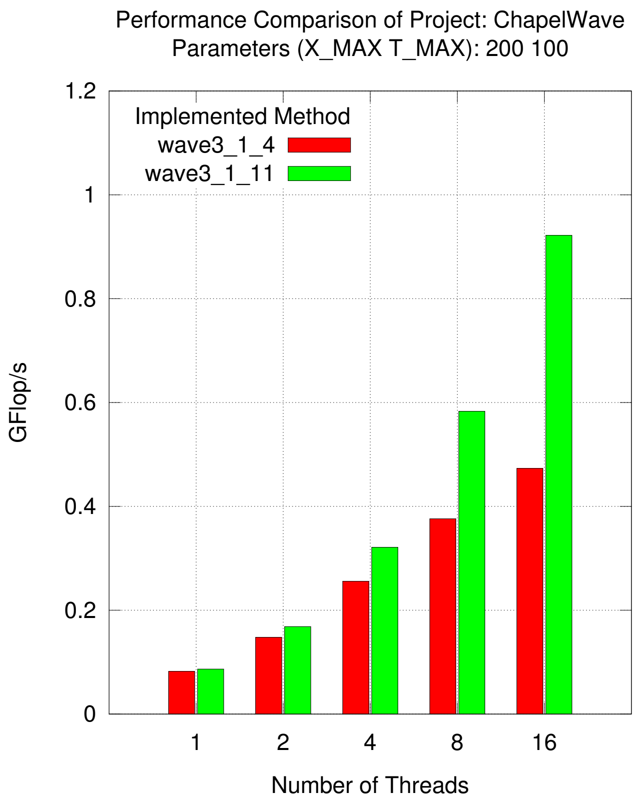
Recomputed results graph.
University of Basel (2015).

[1]*Schnor et al.*: Non-Blocking collectives. Seminar at University of Potsdam

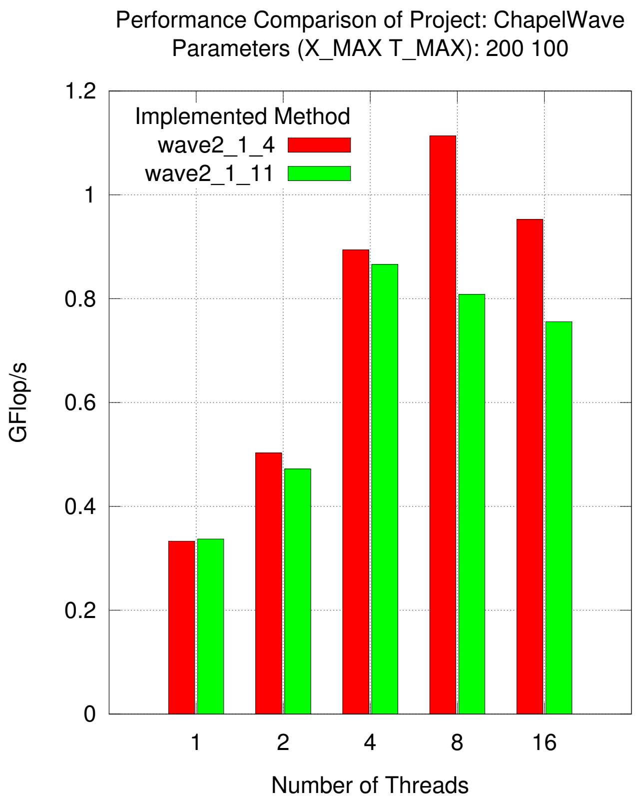
Case Study 3 – Period of time Observation

Experiment definition	2015 University of Basel (HPWC Group) *Original experiment*	2015 University of Basel (HPWC Group) *Recomputation*
Problem	Calculate a 3-D wave equation of 200^3 elements (IEEE double precision arithmetic) in 100 timesteps.	**Unchanged**
System	**SW**: PGAS programming model using Chapel v1.4, GCC 4.7.2 **HW**: 1 node - CPU: 2x AMD Opteron 6274 16-Core, 2.2 GHz, 12 MiB L3 cache - RAM: 256 GiB - OS: Ubuntu 12.04.4, Kernel 3.8.0-38	**SW**: PGAS programming model using Chapel v1.11, GCC 4.7.2 **HW**: **Unchanged**
Method	**1.** Use 3 nested “for” loops (outermost parallelized). **2.** Use “forall” statement with “domain map” function.	**Unchanged**

Case Study 3 – Period of time Observation



Chapel 3D wave "forall with domain map" variant.

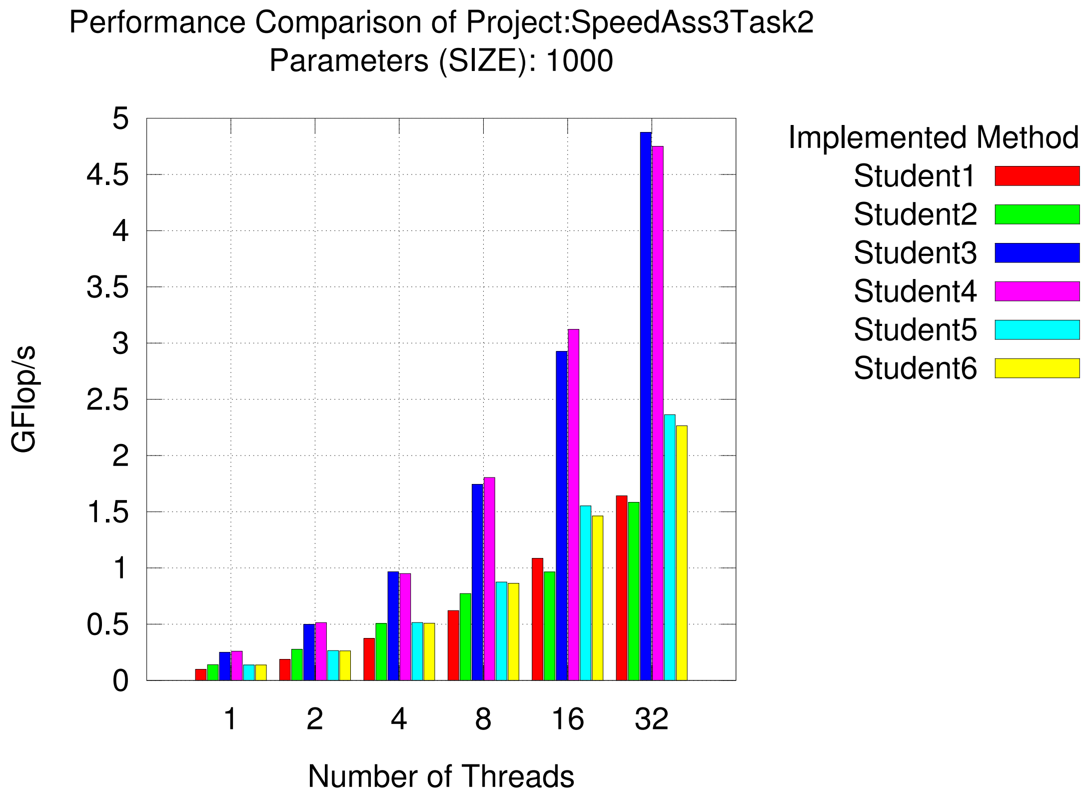


Chapel 3D wave "triply nested loop" variant.

Case Study 4 – Reproducibility in the Curriculum

Experiment definition	2015 **University of Basel (HPWC Student)** *Original experiment*	2015 **University of Basel (HPWC Group)** *Replication*
Problem	Compute a 2-D Matrix Multiplication 1000^2 elements (IEEE double precision arithmetic).	**Unchanged**
System	**SW**: Shared-memory computer using OpenMP 3.1, GCC 4.7.2 **HW**: 1 node - CPU: 2x AMD Opteron 6274 16-Core, 2.2 GHz, 12 MiB L3 cache - RAM: 256 GiB - OS: Ubuntu 12.04.4, Kernel 3.8.0-38	**Unchanged**
Method	**To be defined by students:** - Naive - cache awareness …	**Unchanged**

Case Study 4 – Reproducibility in the Curriculum



Students' assignment Replication. Performance comparison.

Case Study 4 – Reproducibility in the Curriculum

Student 3 and 4 claim to get the best performance. Can we trust them?

A live test!

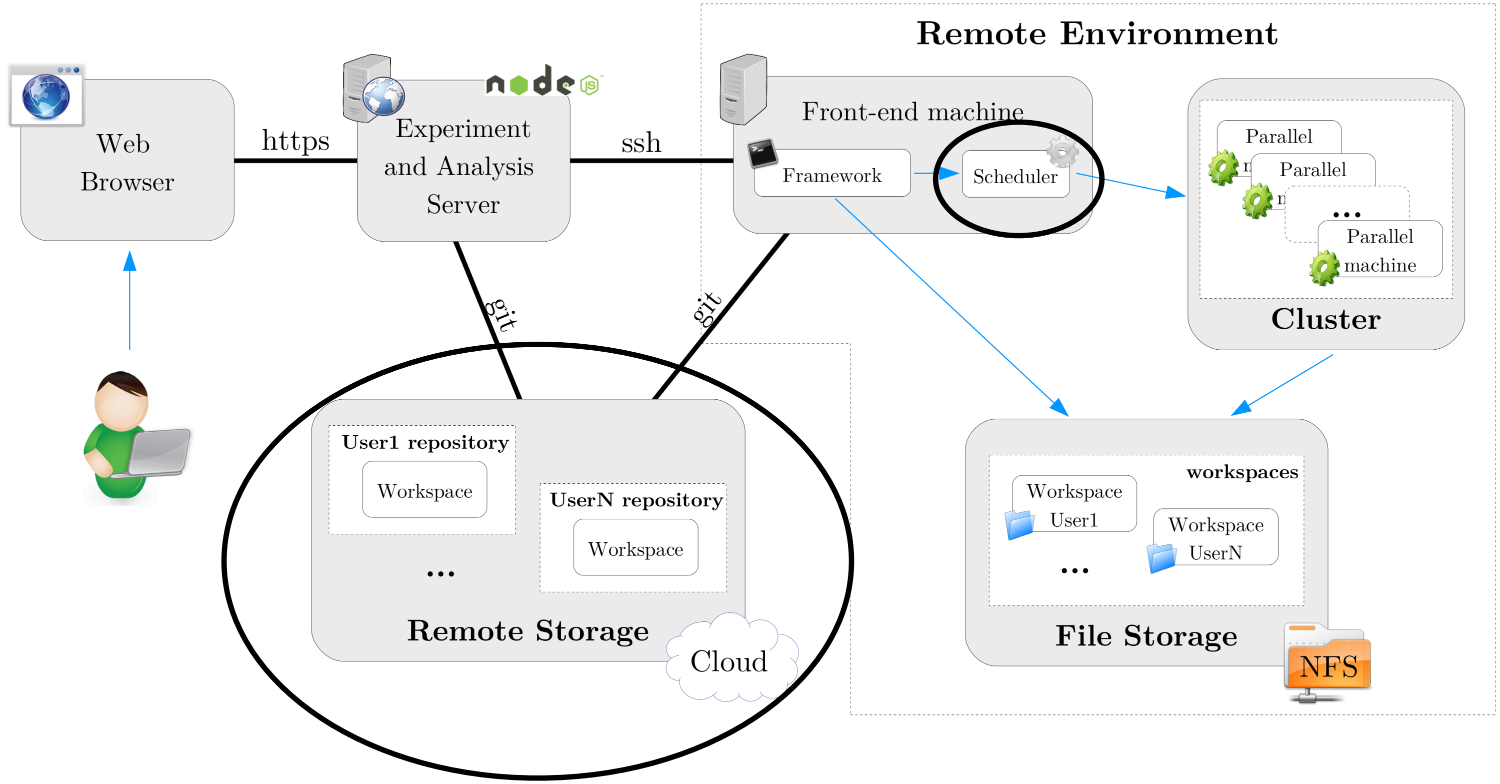
Conclusions

- Repeatability of an experiment only possible if **precise description of experiment** is given: Problem, System, and Method.
- **Repeatability terminology** needs to be sharpened.
- **Replicability**: World-wide access to experiments through Internet feasible (security and authentication mechanisms essential).
- **Recomputation and reproducibility:** Harder to achieve but not impossible.
- **Collaboration support** for performance engineering needed.
- **Integration into the curriculum:** The next generation can/must do better.

Future work

– **Collaborative Performance Engineering**

– **Porting to external HPC environments**



- Maia
- Erlangen
- CSCS
- ...

– **Third-party software for managing environment (ex. EasyBuild)**
– **Correctness check for experiment’s results**
– **Better visualization of experiments and results**

University of Basel

Thank you

for your attention.