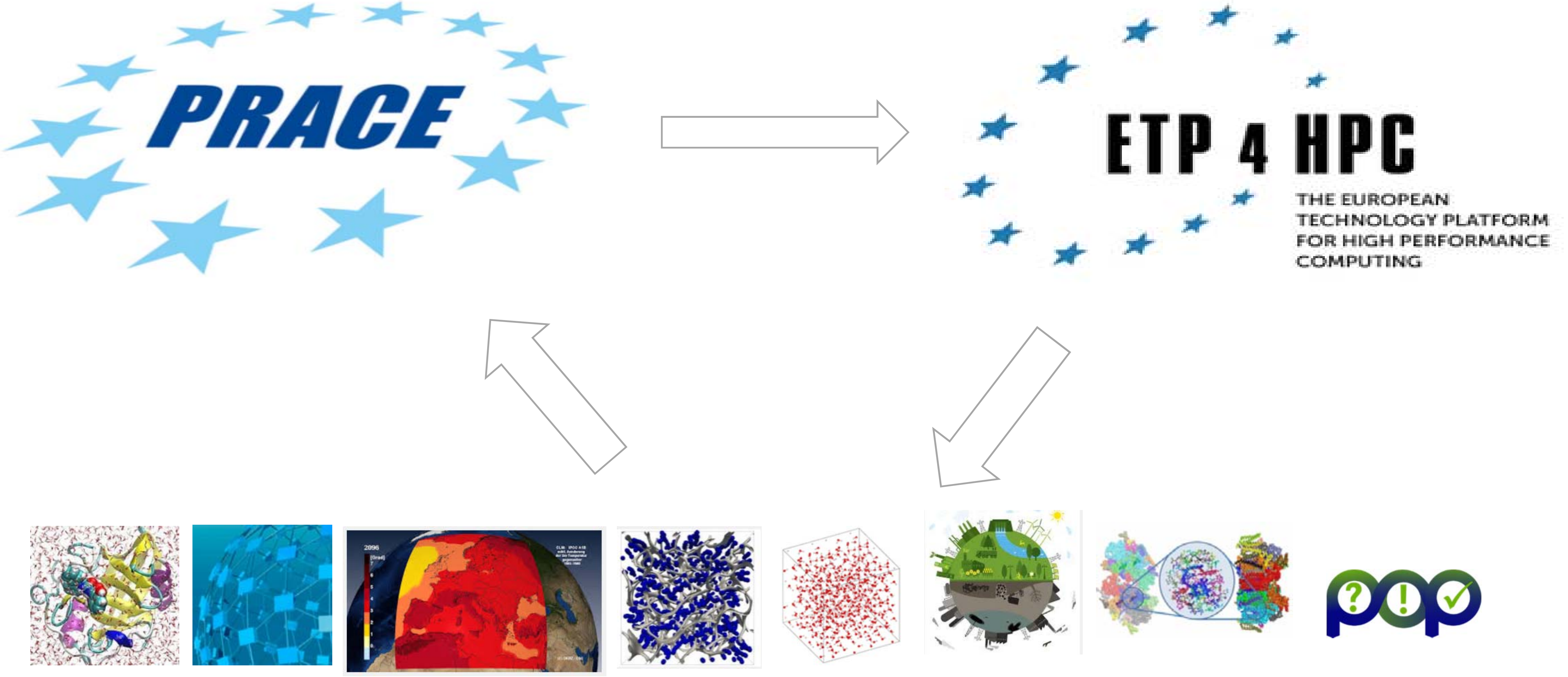
The POP Project

Jesús Labarta (BSC)

6th AICS Symposium

Kobe, Feb 23rd, 2016

EU HPC Ecosystem


PRACE
ETP 4 HPC
THE EUROPEAN TECHNOLOGY PLATFORM FOR HIGH PERFORMANCE COMPUTING

Centres of Excellence in **HPC applications**

POP CoE

- A **Center of Excellence**
  - On **Performance Optimization and Productivity**
  - Promoting **best practices in parallel programming**
- Providing **Services**
  - Precise understanding of application and system behavior
  - Suggestion/support on how to refactor code in the most productive way
- **Horizontal**
  - Transversal across application areas, platforms, scales
- **For academic AND industrial codes and users !**

Motivation

Why?

- Complexity of machines and codes

  → Frequent lack of quantified understanding of actual behavior

  → Not clear most productive direction of code refactoring

- Important to maximize
  - Efficiency (performance, power) of compute intensive applications
  - Productivity of the development efforts

Partners

- **Who?**
  - BSC (coordinator), ES
  - HLRS, DE
  - JSC, DE
  - NAG, UK
  - RWTH Aachen, IT Center, DE
  - TERATEC, FR

A team with

- Excellence in performance tools and tuning
- Excellence in programming models and practices
- Research and development background AND
  proven commitment in application to real academic and industrial use cases

Services provided by the CoE

? Application Performance Audit

- Primary service
- Identify performance issues of customer code (at customer site)
- Small effort (< 1 month)

! Application Performance Plan

- Follow-up on the audit service
- Identifies the root causes of the issues found and qualifies and quantifies approaches to address them
- Longer effort (1-3 months)

✓ Proof-of-Concept

- Experiments and mock-up tests for customer codes
- Kernel extraction, parallelization, mini-apps experiments to show effect of proposed optimizations
- 6 months effort

Target customers

- **Code developers**
  - Assessment of detailed actual behavior
  - Suggestion of most productive directions to refactor code
- **Users**
  - Assessment of achieved performance in specific production conditions
  - Possible improvements modifying environment setup
  - Evidence to interact with code provider
- **Infrastructure operators**
  - Assessment of achieved performance
    in production conditions
  - Possible improvements from modifying environment setup
  - Information for computer time allocation processes
  - Training of support staff
- **Vendors**
  - Benchmarking
  - Customer support
  - System dimensioning/design

About methodologies

Target and approach

- Real production codes and operation conditions
- Install tools @ customer production machine (local, PRACE, …)
- Interactively: Gather data → Analysis → Report

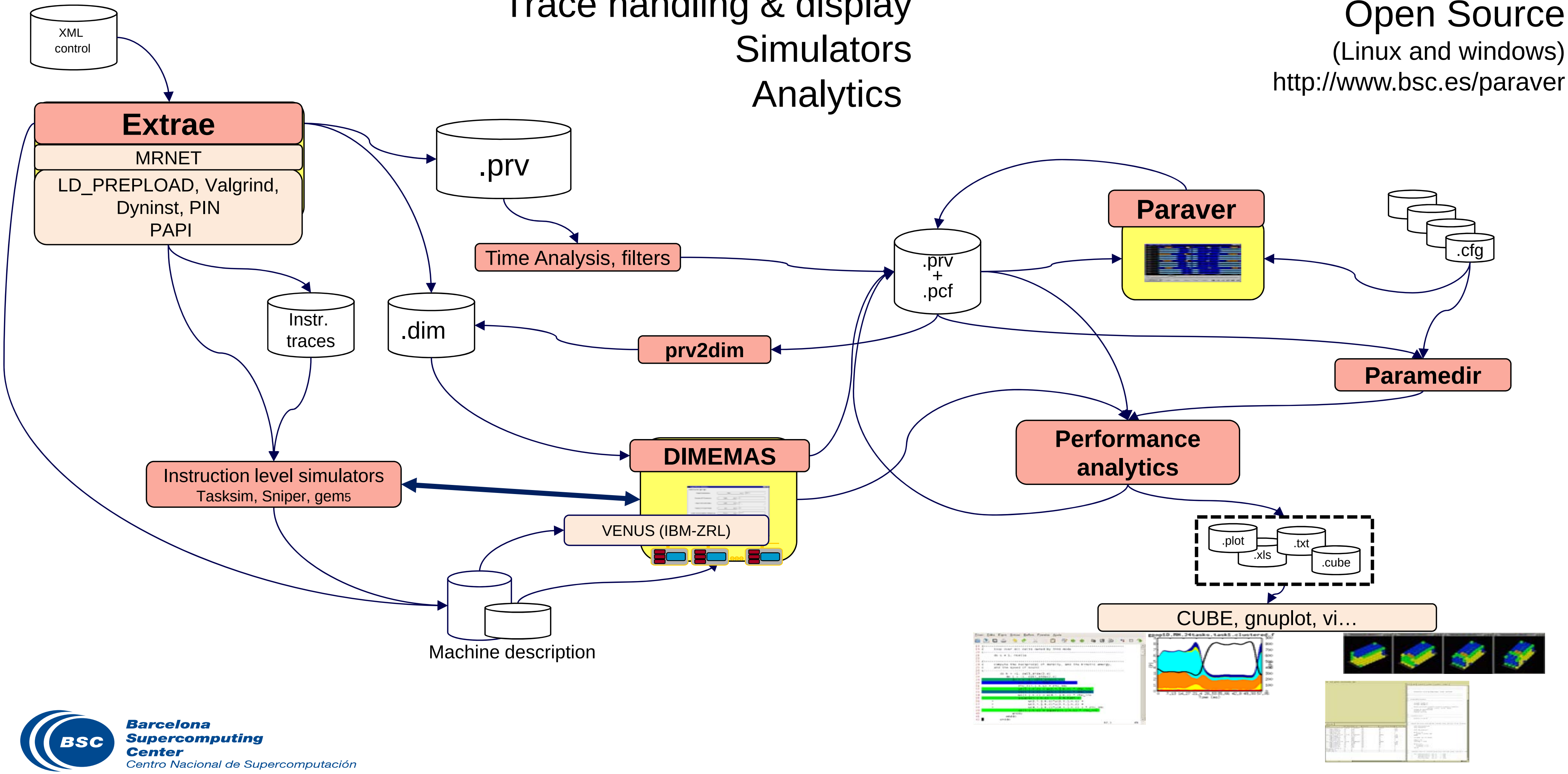
Challenge

- Integration of methodologies
  - How to look at performance in a hierarchical/structured way
  - Tools to validate/reject hypotheses and help generate new ones
- Duration of studies?

BSC – tools framework

Trace handling & display
Simulators
Analytics

Open Source
(Linux and windows)
http://www.bsc.es/paraver



XML control

Extrae
MRNET
LD_PRELOAD, Valgrind, Dyninst, PIN
PAPI

.prv

Time Analysis, filters

.prv + .pcf

Paraver

.cfg

Instr. traces

.dim

prv2dim

Paramedir

DIMEMAS

Performance analytics

Instruction level simulators
Tasksim, Sniper, gems

VENUS (IBM-ZRL)

.plot
.xls
.txt
.cube

CUBE, gnuplot, vi…

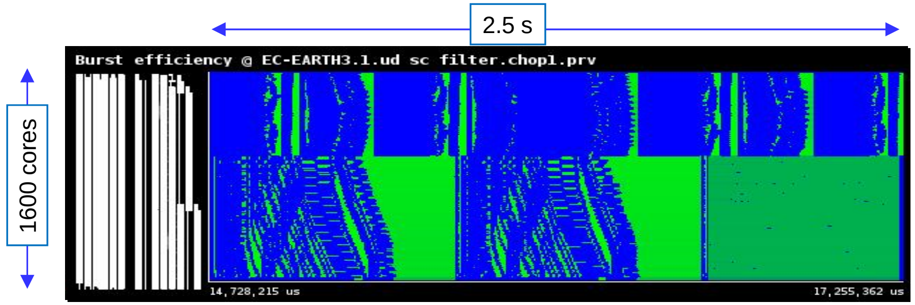
Machine description

Barcelona
Supercomputing
Center
Centro Nacional de Supercomputación

BSC Performance Tools

Flexible trace visualization and analysis

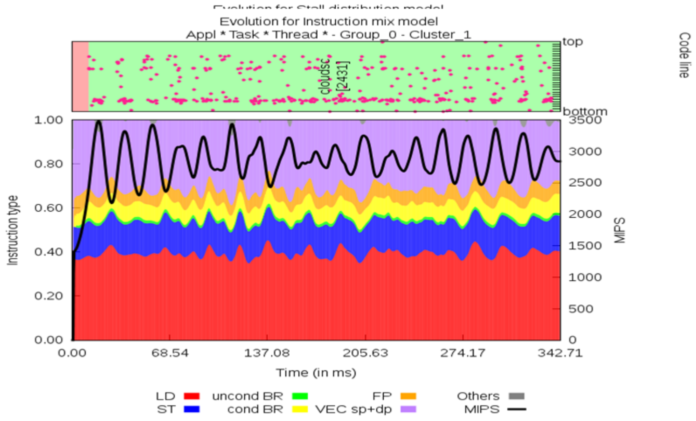
Adaptive burst mode tracing



BSC-ES – EC-EARTH

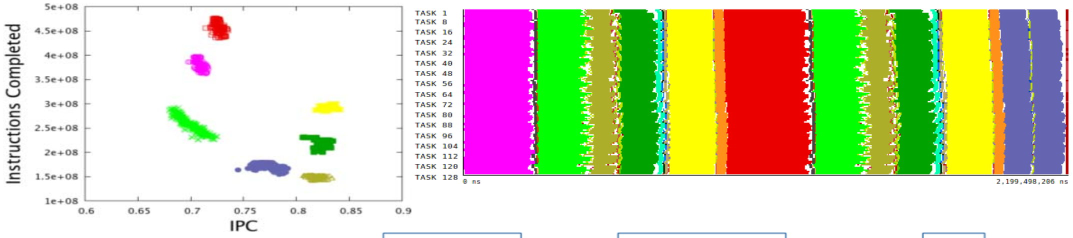
26.7MB trace
Eff: 0.43; LB: 0.52; Comm:0.81

Instantaneous metrics for ALL hardware counters at “no” cost



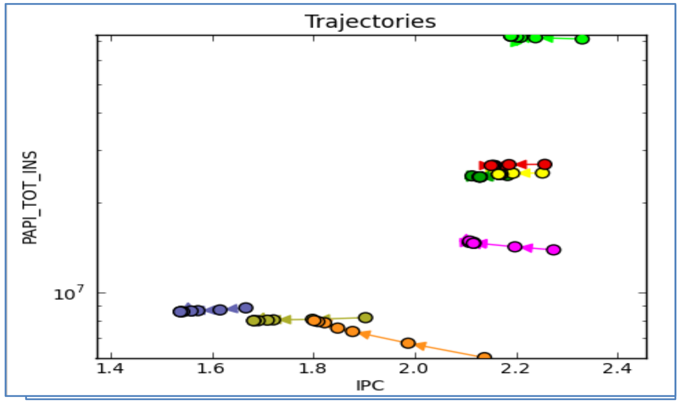
BSC-ES – EC-EARTH

Advanced clustering algorithms



Instruction mix
Memory hierarchy
Stalls

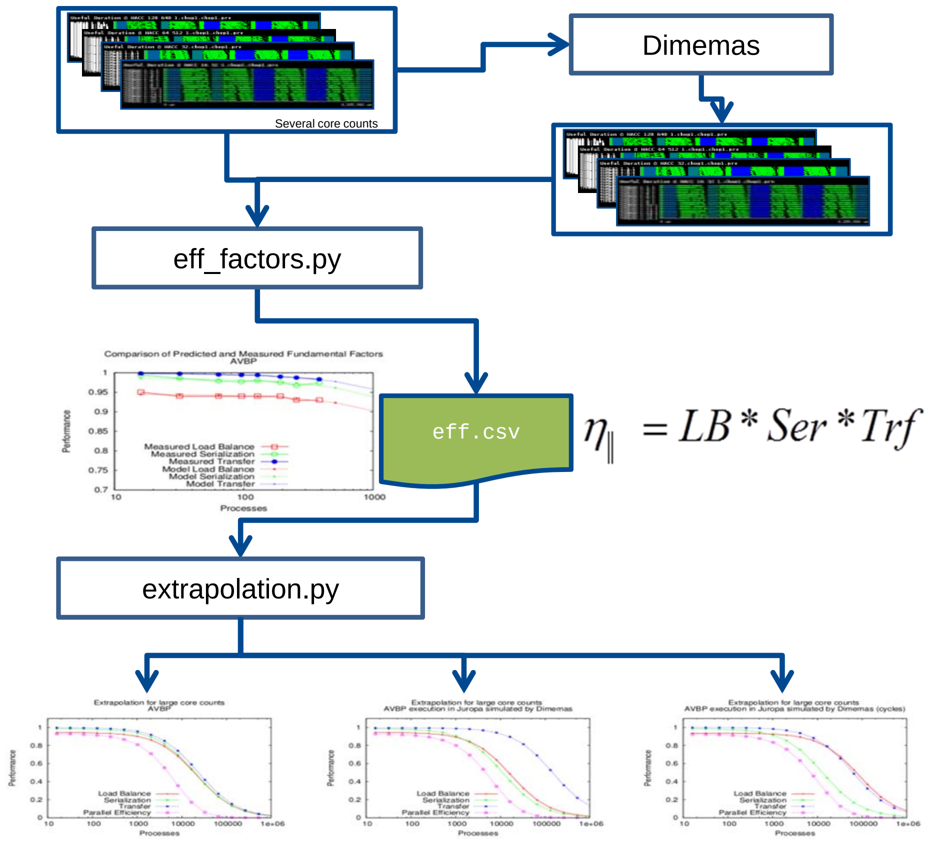
Tracking performance evolution



AMG2013

BSC Performance Tools

Models and Projection



$$\eta_{\parallel} = LB * Ser * Trf$$

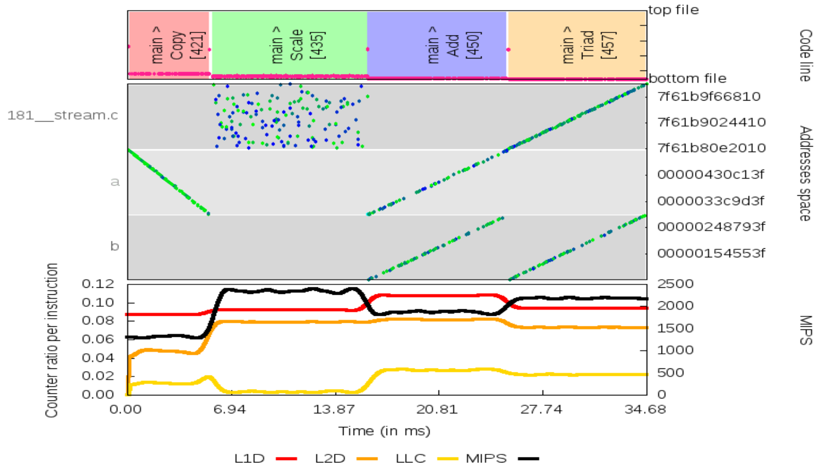
No MPI noise

+ No OS noise

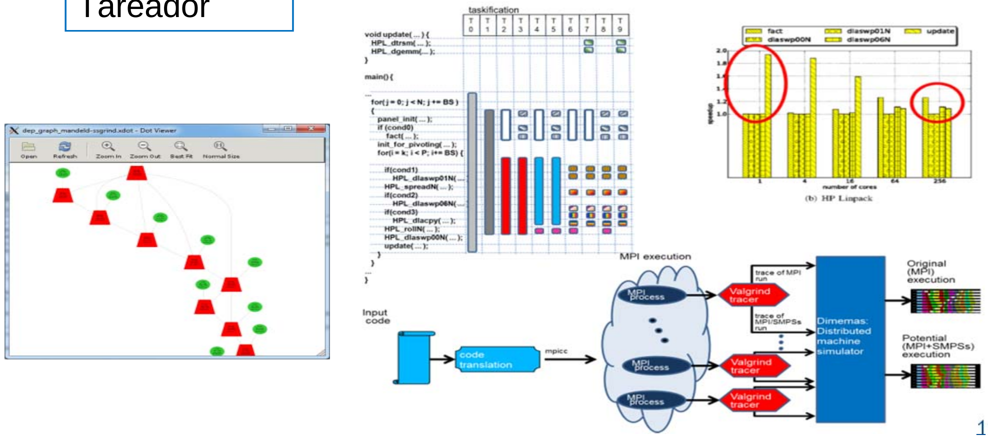
"Scalability prediction for fundamental performance factors " J. Labarta et al. SuperFRI 2014

Intel –BSC Exascale Lab

Data access patterns



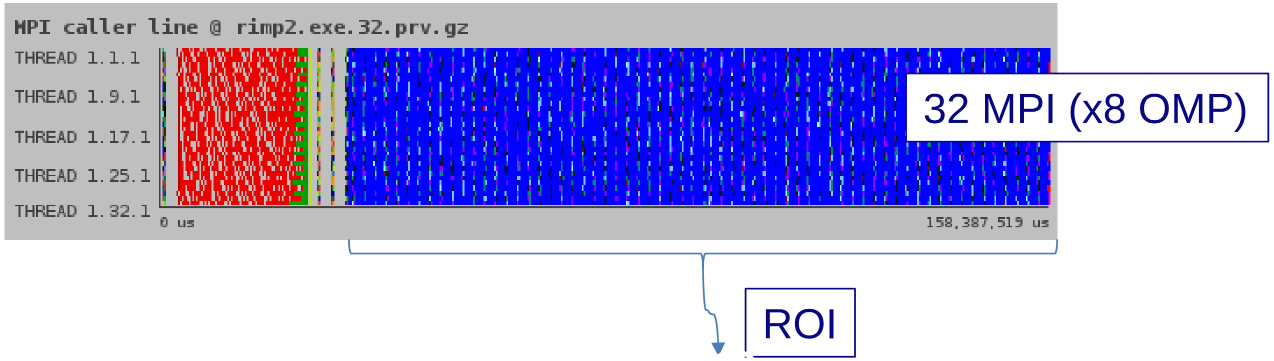
Tareador



An example ntchem-mini (taxol)

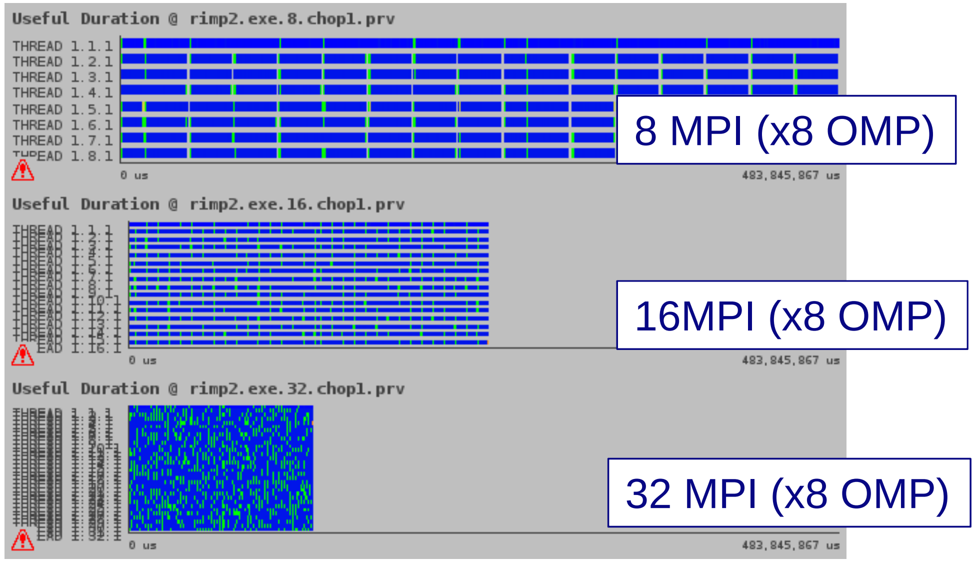
Structure

- Syntactic structure
- Spatio – temporal structure
  - Computation rather than MPI
  - Towards fundamental parallel behavior and concepts



32 MPI (x8 OMP)

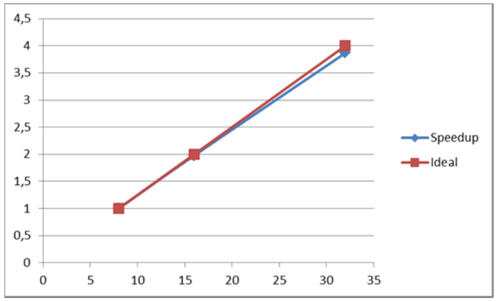
ROI



8 MPI (x8 OMP)

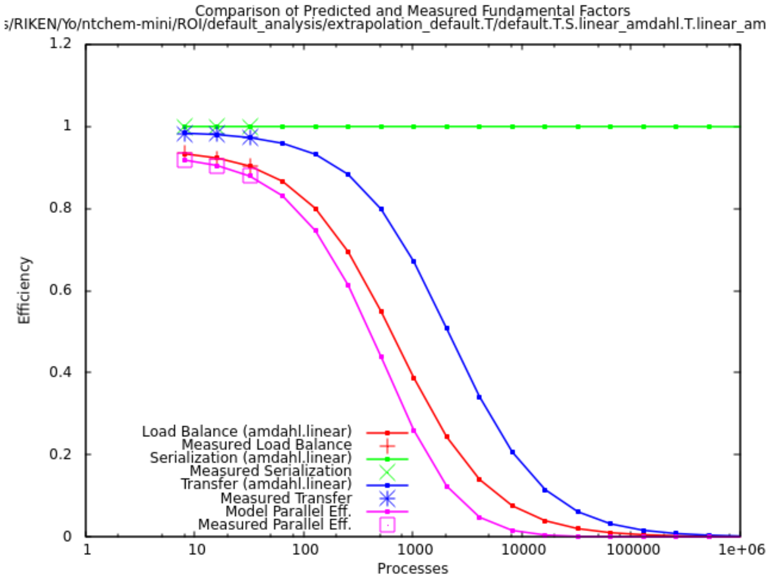
16MPI (x8 OMP)

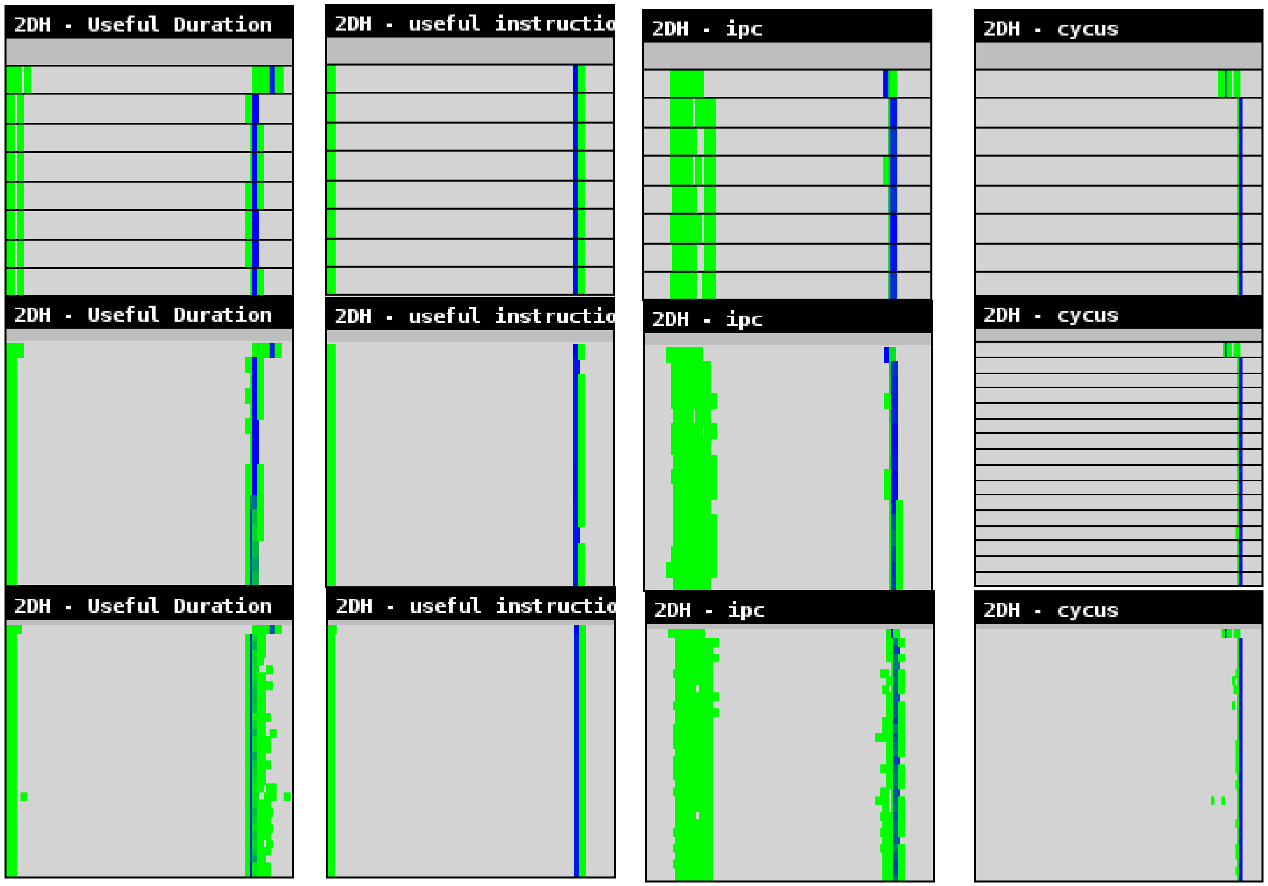
32 MPI (x8 OMP)



Efficiency factors and extrapolation

$$\eta_{\parallel} = LB * Ser * Trf$$





Time imbalance … on first process !!!
Same granularity (~7ms) for all core counts !!

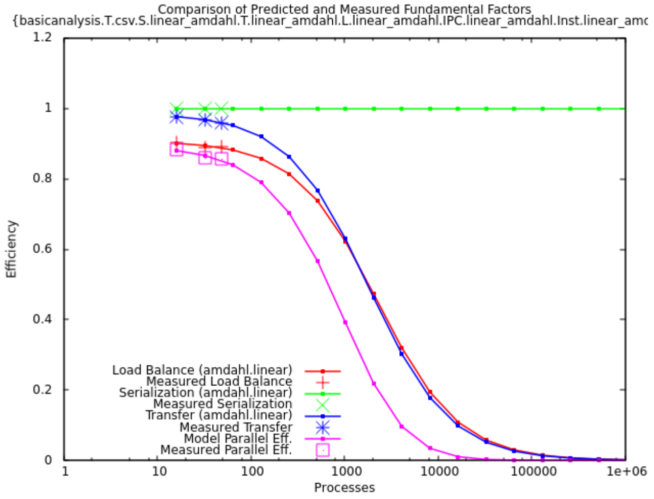
Some IPC imbalance

Computationally balanced!!

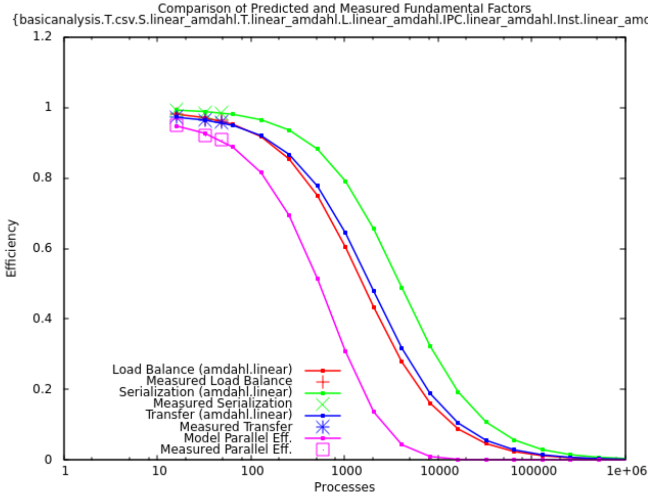
Some “frequency” imbalance

Addressing fundamental bottlenecks

- Avoiding first process imbalance
  - Compiling without DEBUG define
  - Using Dimemas and cycle based trace conversion

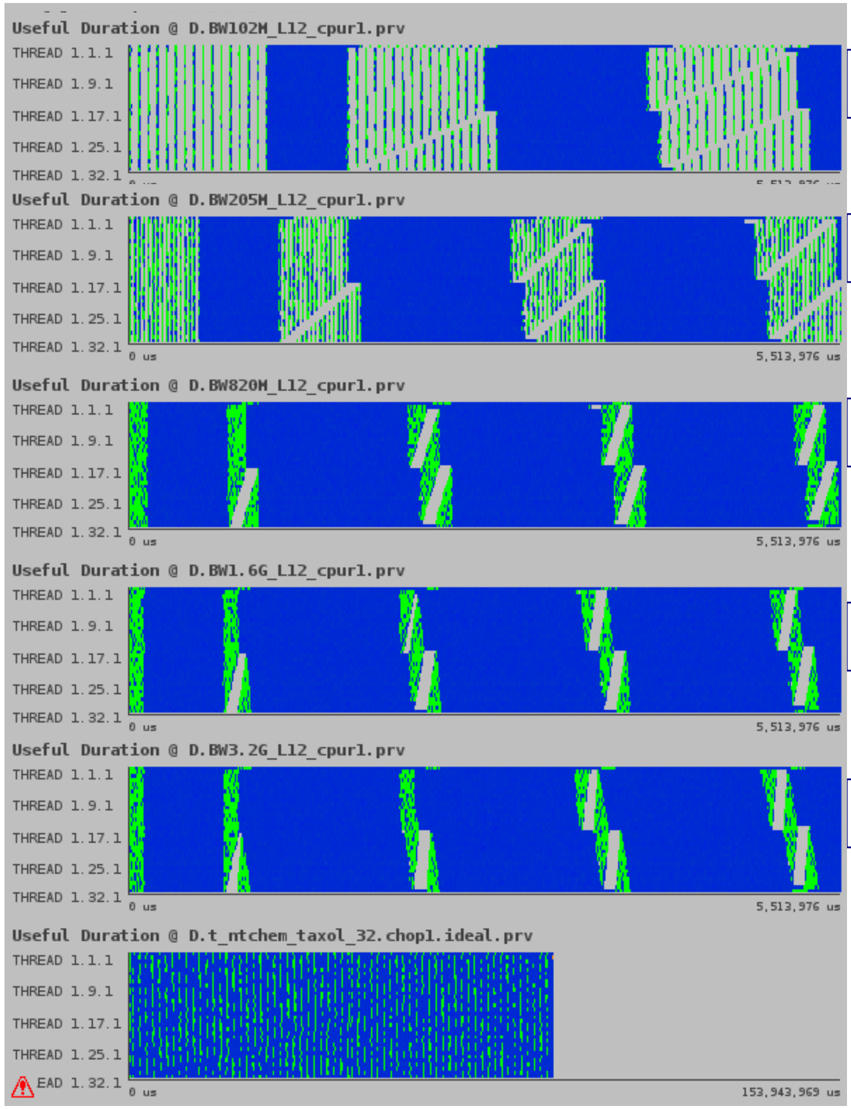


- Fair improvement in load balance …. but still !!!
- Transfer stays very important !!!
- Serialization effects appeared !!!



Transfer impact ?

- Impact of bandwidth
  - 7.5 MB msgs



BW / 4

BW / 2

BW

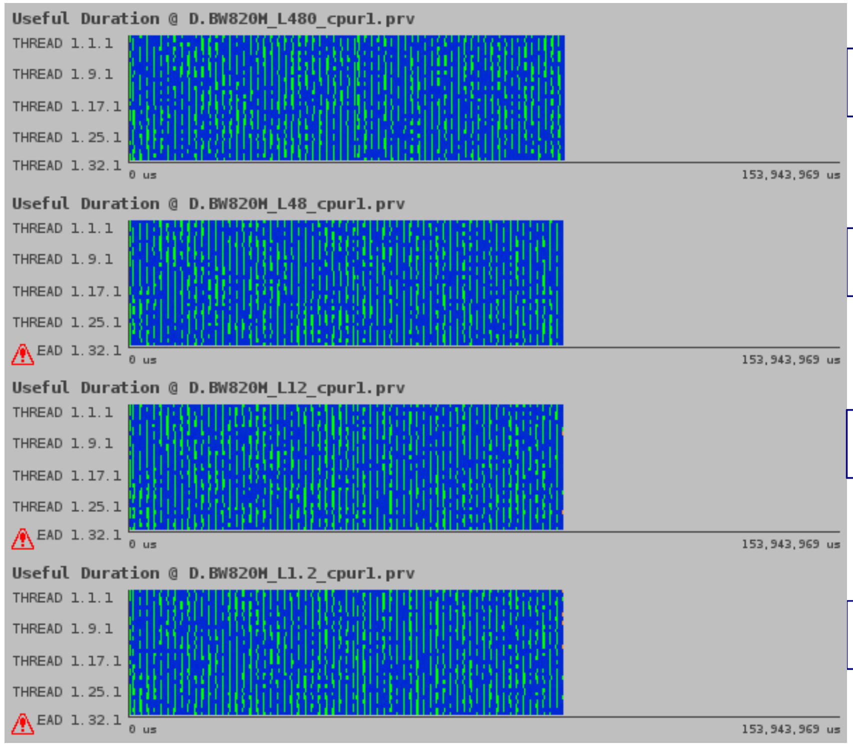
BW x 2

BW x 4

BW ideal

Transfer impact ?

- Impact of Latency
  - Not relevant → would splitting messages give opportunity for overlap?



Useful Duration @ D.BW820M_L480_cpurl.prv

Lat * 40

Useful Duration @ D.BW820M_L48_cpurl.prv

Lat * 4

Useful Duration @ D.BW820M_L12_cpurl.prv

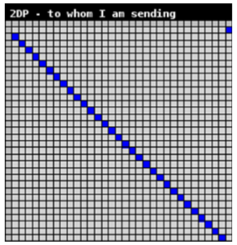
Lat

Useful Duration @ D.BW820M_L1.2_cpurl.prv

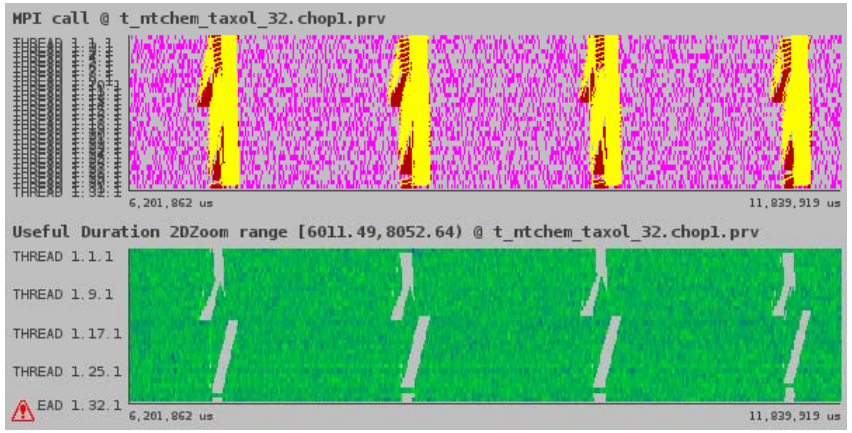
Lat / 10

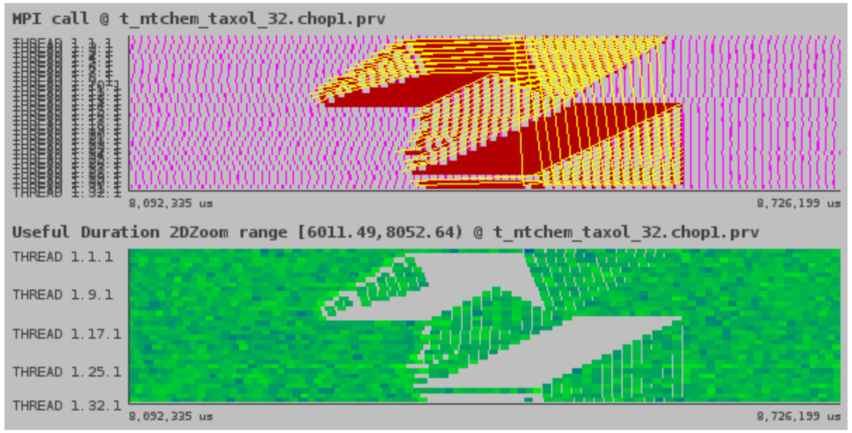
Transfer impact ?

- Lots of Collective communications separating fixed grain computation bursts
  - Communicator size == 1
  - Minimal cost



- Rotating communication pattern
  - Paying imbalance
  - Can be better balanced?
  - Could benefit from DLB?
- Can communication be scattered?
  - Overlap to tolerate BW shortage?

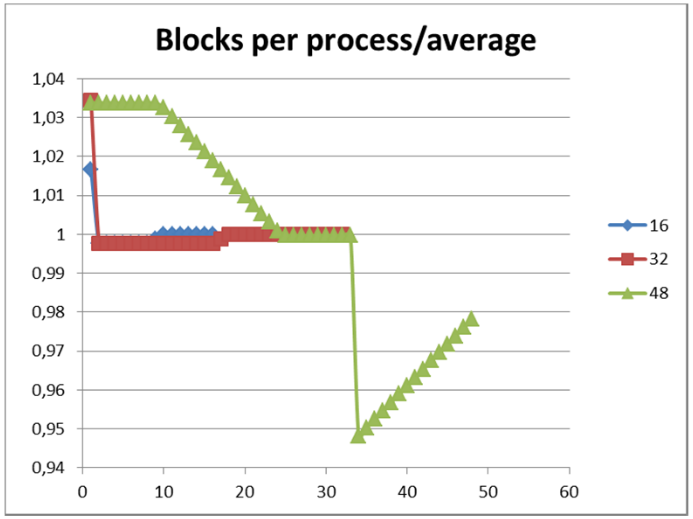

MPI call @ t_ntchem_taxol_32.chop1.prv
6,201,862 us
11,839,919 us
Useful Duration 2DZoom range [6011.49,8052.64) @ t_ntchem_taxol_32.chop1.prv
THREAD 1.1.1
THREAD 1.9.1
THREAD 1.17.1
THREAD 1.25.1
EAD 1.32.1
6,201,862 us
11,839,919 us

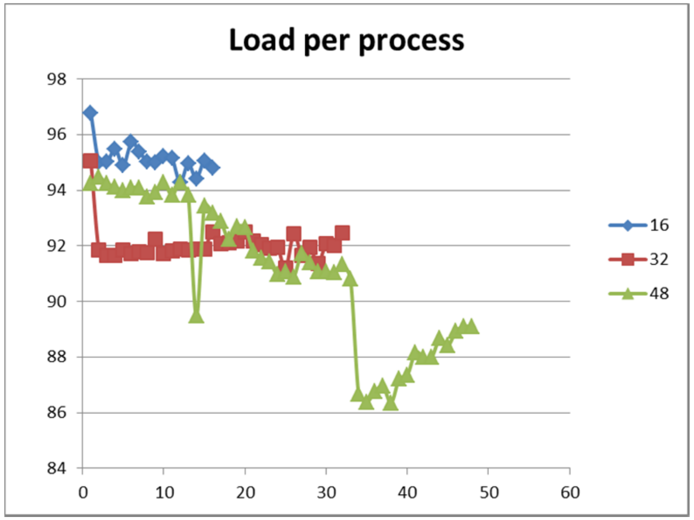

MPI call @ t_ntchem_taxol_32.chop1.prv
8,092,335 us
8,726,199 us
Useful Duration 2DZoom range [6011.49,8052.64) @ t_ntchem_taxol_32.chop1.prv
THREAD 1.1.1
THREAD 1.9.1
THREAD 1.17.1
THREAD 1.25.1
THREAD 1.32.1
8,092,335 us
8,726,199 us

Why imbalance?

- Same granularity at all core counts
- Strong scaling

- → Same total number of “grains” (402753) split over processes



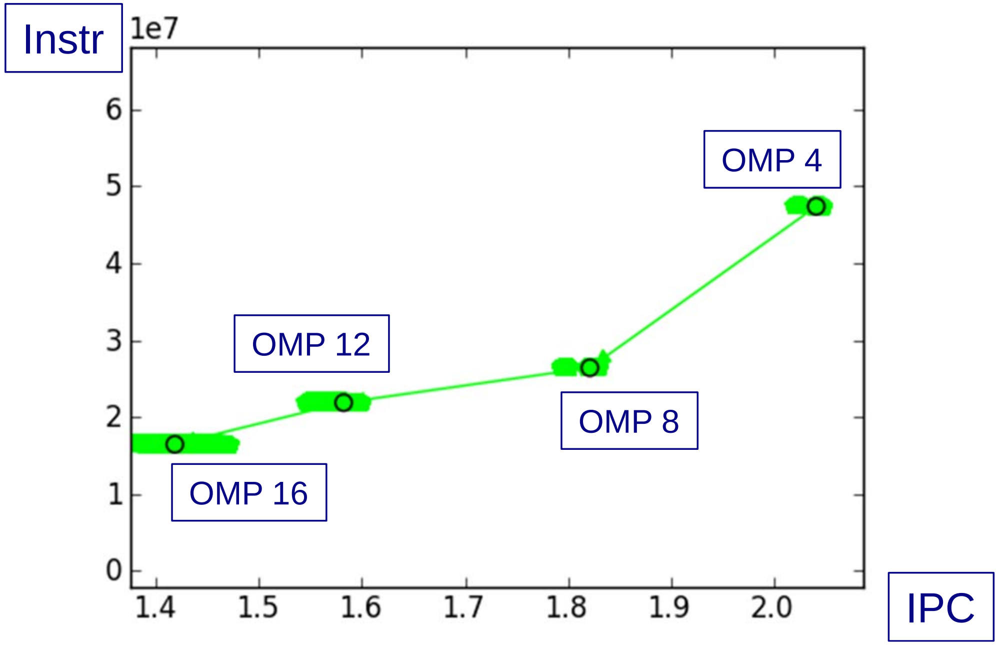


Barcelona Supercomputing Center
Centro Nacional de Supercomputación

OpenMP Behavior?

- Even if the trace is of only one thread per process?

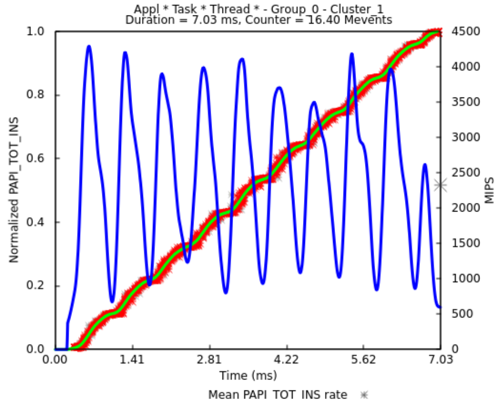
- Inference
  - IPC impact
    - Memory BW? Invalidations? …
  - Some serial parts
  - Variability



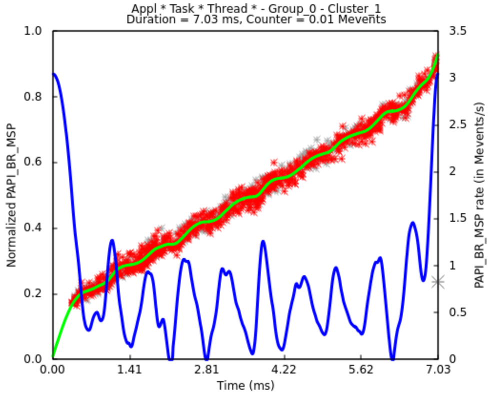
Fine grain behavior?

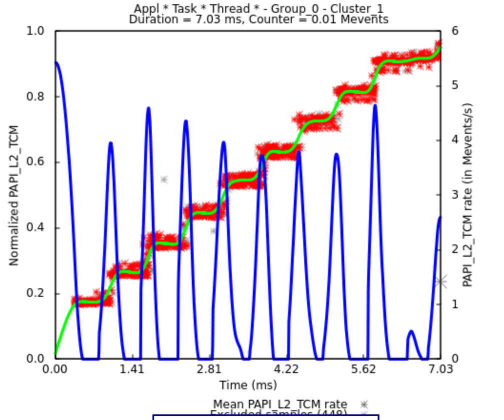
- Iterative internal structure?
  - 10 “iterations”
- L2 Cache misses
  - @ transition
  - Not within “iteration”
- Opportunities
  - Block prefetch ?
  - Nested parallelism?

GIPS

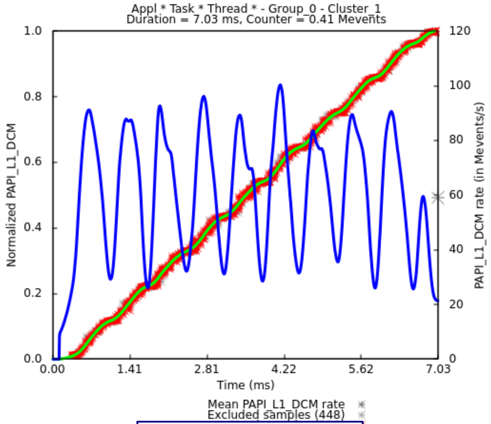


Branch mispredicts





L2 misses



L1 misses

Conclusion

POP

- develop best practices
- integrate analysis methodologies
- Towards accelerating the analysis cycle
- Increasing insight and guidance to application developer

Opportunities

- Apply to other applications
- And target machines
- Integrate other tools