www.bsc.es

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Understanding applications with Paraver

Judit Gimenez
judit@bsc.es

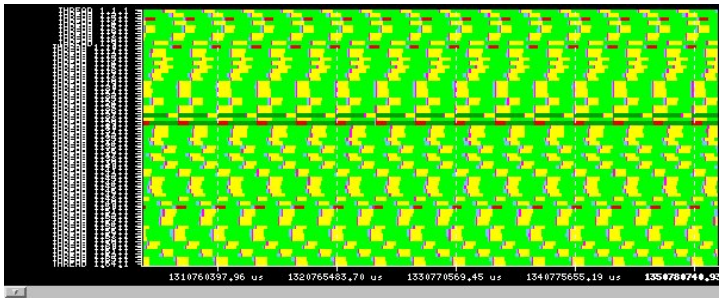
NERSC - Berkeley, August 2016

Humans are visual creatures

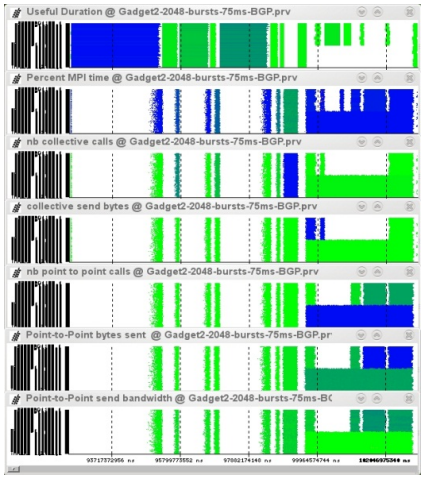
- Films or books? PROCESS
  - Two hours vs. days (months)
- Memorizing a deck of playing cards STORE
  - Each card translated to an image (person, action, location)
- Our brain loves pattern recognition IDENTIFY
  - What do you see on the pictures?

Our Tools

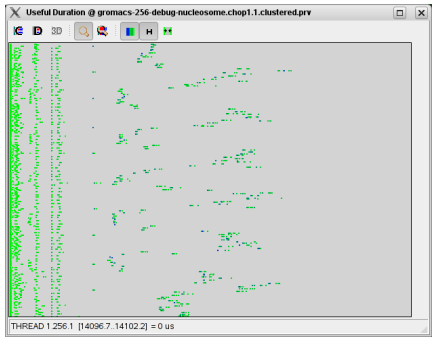
- Since 1991
- Based on traces
- Open Source
  - http://www.bsc.es/paraver



- Core tools:
  - Paraver (paramedir) – offline trace analysis
  - Dimemas – message passing simulator
  - Extrae – instrumentation



- Focus
  - Detail, variability, flexibility
  - Behavioral structure vs. syntactic structure
  - Intelligence: Performance Analytics



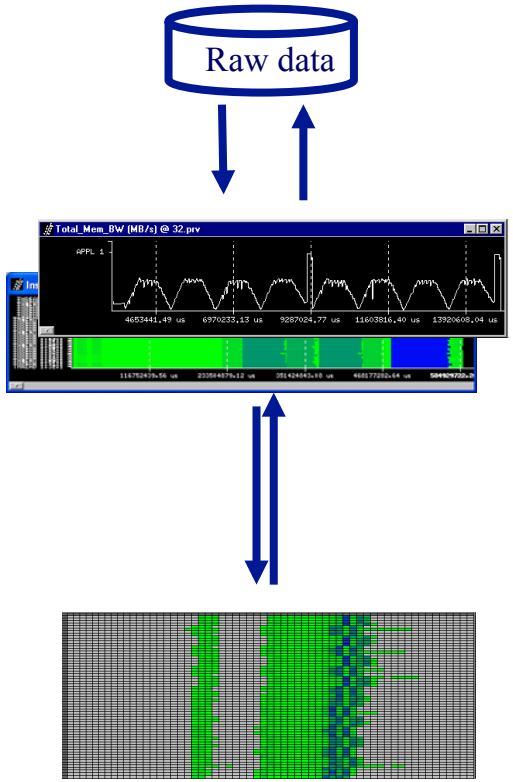
www.bsc.es

Barcelona Supercomputing Center

Centro Nacional de Supercomputación

Paraver

Paraver – Performance data browser



Timelines

2/3D tables (Statistics)

Trace visualization/analysis

+ trace manipulation

Goal = Flexibility

No semantics

Programmable

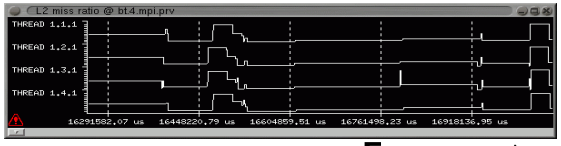
Comparative analyses

Multiple traces

Synchronize scales

Timelines

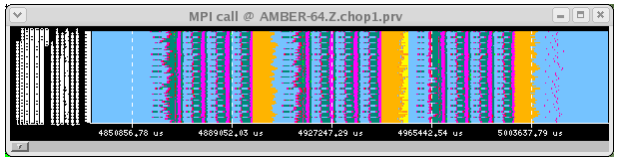
- Each window displays one view
  - **Piecewise constant** function of time



$$s(t) = S_i, i \in [t_i, t_{i+1})$$

- Types of functions
  - Categorical
    - State, user function, outlined routine

$$S_i \in [0, \mathrm{n}] \subset N, \quad n <$$

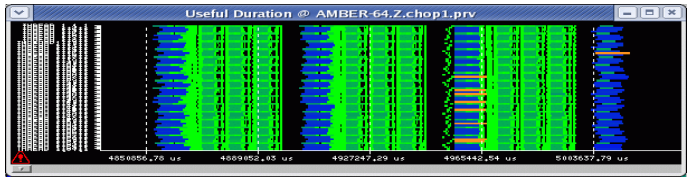


  - Logical
    - In specific user function, In MPI call, In long MPI call

$$S_i \in \{0, 1\}$$

  - Numerical
    - IPC, L2 miss ratio, Duration of MPI call, duration of computation burst

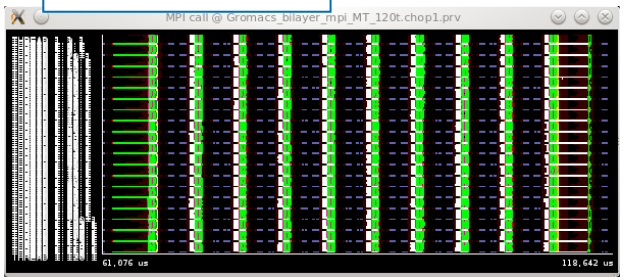
$$S_i \in R$$



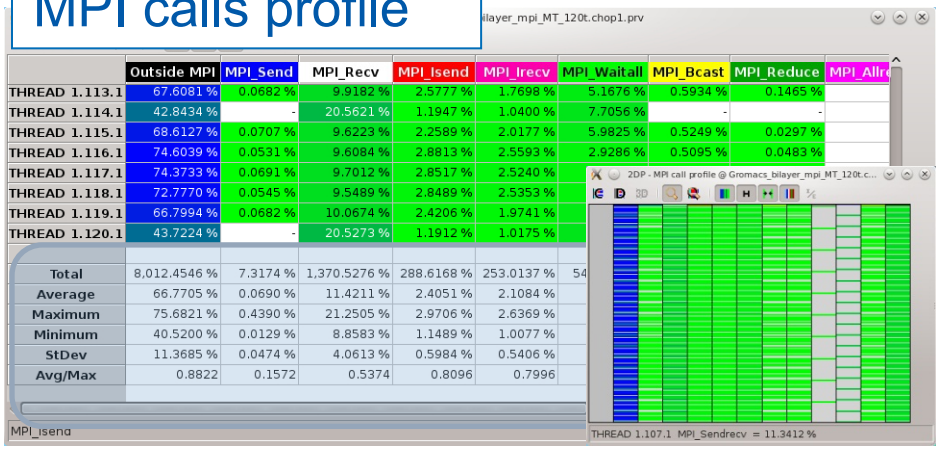
Tables: Profiles, histograms, correlations

From timelines to tables

MPI calls

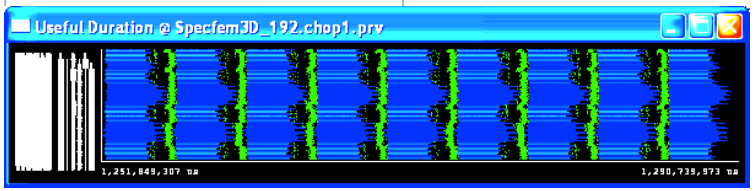


MPI calls profile

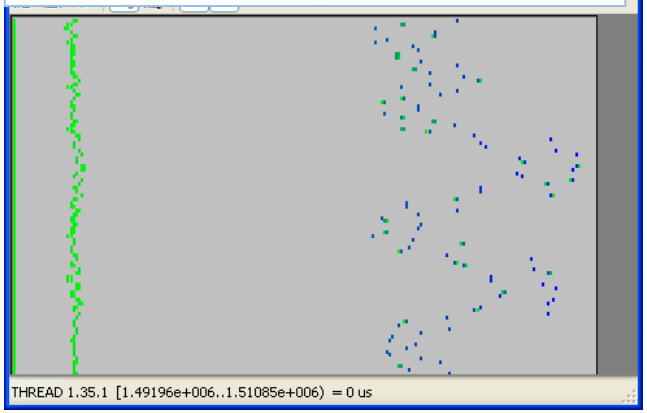


	Outside MPI	MPI_Send	MPI_Recv	MPI_Isend	MPI_Irecv	MPI_Waitall	MPI_Bcast	MPI_Reduce	MPI_Allr
THREAD 1.113.1	67.6081 %	0.0682 %	9.9182 %	2.5777 %	1.7698 %	5.1676 %	0.5934 %	0.1465 %	
THREAD 1.114.1	42.8434 %	-	20.5621 %	1.1947 %	1.0400 %	7.7056 %	-	-	
THREAD 1.115.1	68.6127 %	0.0707 %	9.6223 %	2.2589 %	2.0177 %	5.9825 %	0.5249 %	0.0297 %	
THREAD 1.116.1	74.6039 %	0.0531 %	9.6084 %	2.8813 %	2.5593 %	2.9286 %	0.5095 %	0.0483 %	
THREAD 1.117.1	74.3733 %	0.0691 %	9.7012 %	2.8517 %	2.5240 %				
THREAD 1.118.1	72.7770 %	0.0545 %	9.5489 %	2.8489 %	2.5353 %				
THREAD 1.119.1	66.7994 %	0.0682 %	10.0674 %	2.4206 %	1.9741 %				
THREAD 1.120.1	43.7224 %	-	20.5273 %	1.1912 %	1.0175 %				
Total	8,012.4546 %	7.3174 %	1,370.5276 %	288.6168 %	253.0137 %	54			
Average	66.7705 %	0.0690 %	11.4211 %	2.4051 %	2.1084 %				
Maximum	75.6821 %	0.4390 %	21.2505 %	2.9706 %	2.6369 %				
Minimum	40.5200 %	0.0129 %	8.8583 %	1.1489 %	1.0077 %				
StDev	11.3685 %	0.0474 %	4.0613 %	0.5984 %	0.5406 %				
Avg/Max	0.8822	0.1572	0.5374	0.8096	0.7996				

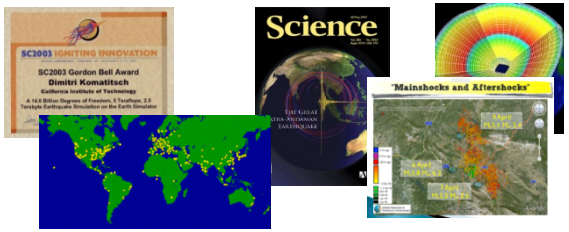
Useful Duration

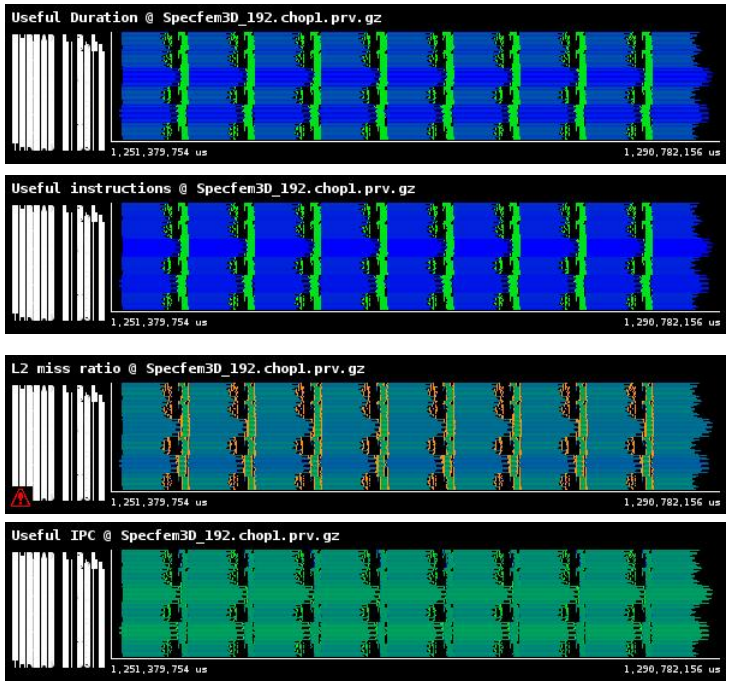


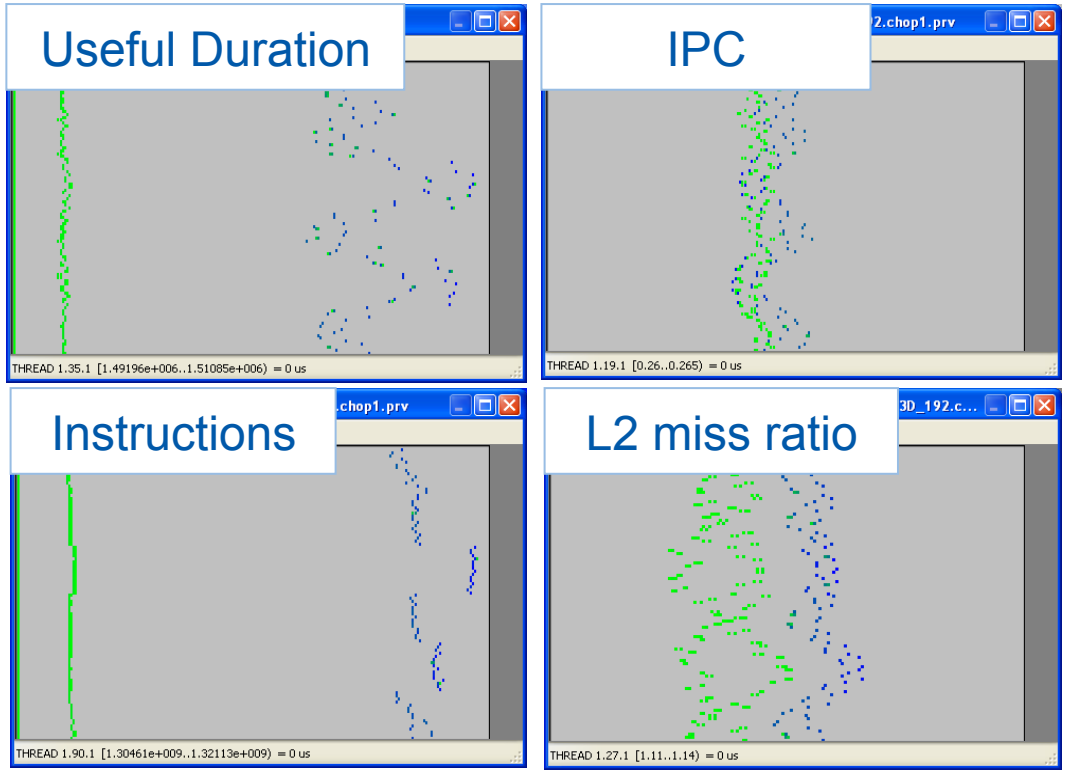
Histogram Useful Duration



Analyzing variability through histograms and timelines

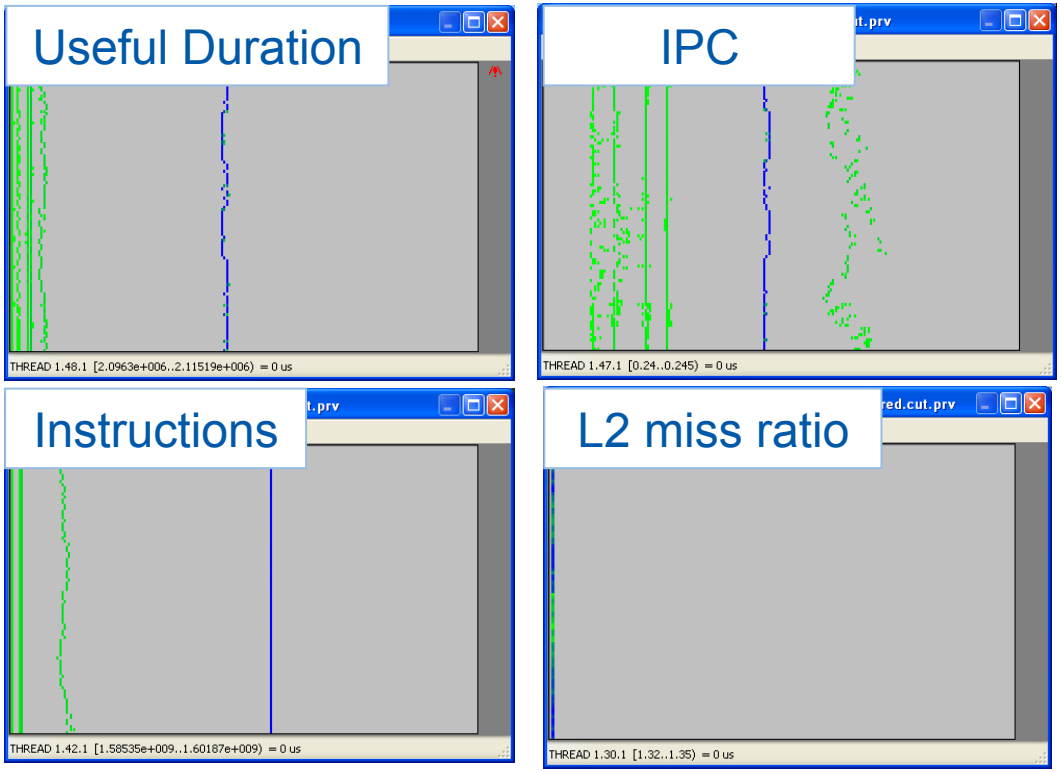






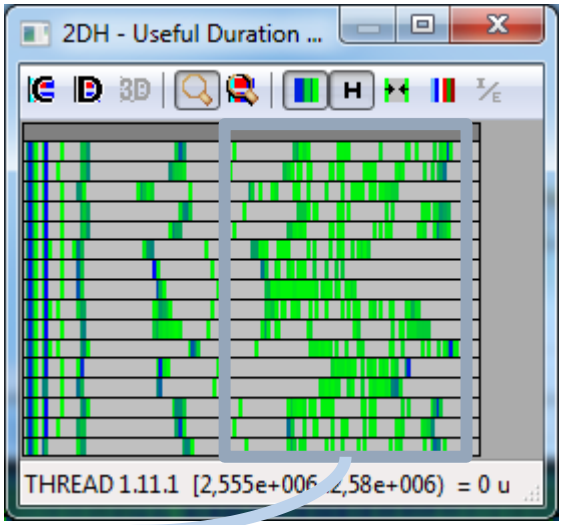
Analyzing variability through histograms and timelines

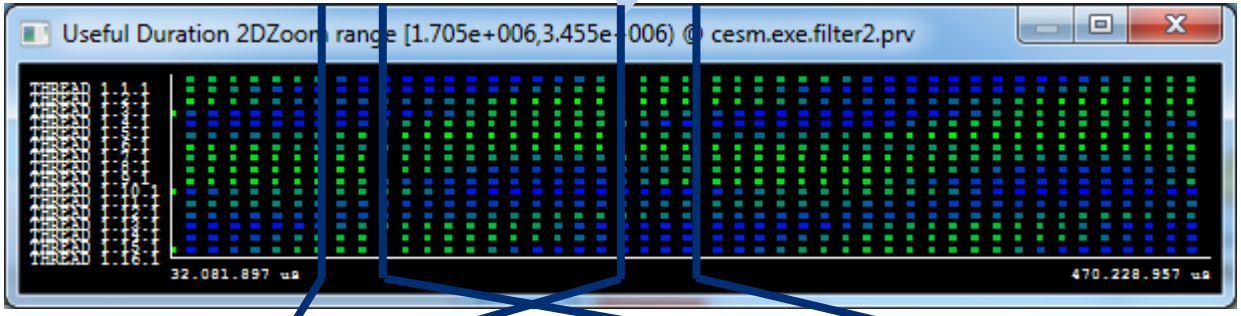
- By the way: six months later ….

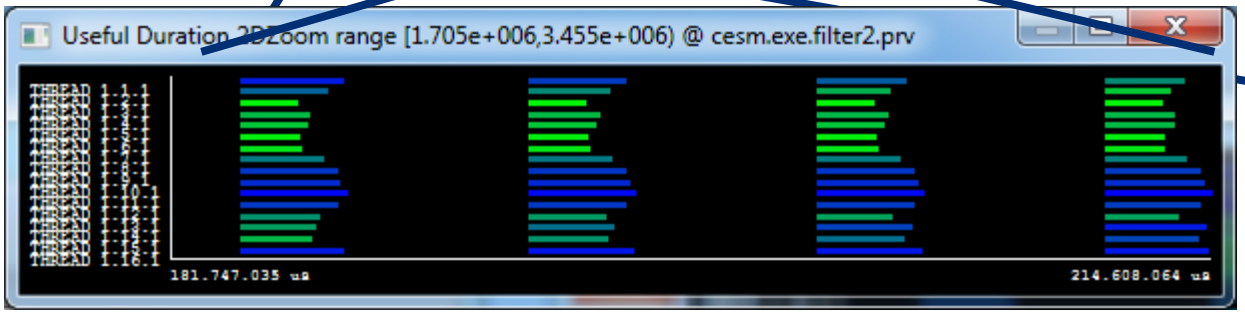


Variability … is everywhere

- CESM: 16 processes, 2 simulated days
- Histogram useful computation duration shows high variability
- How is it distributed?
- Dynamic imbalance
  - In space and time
  - Day and night.
  - Season ? ☺

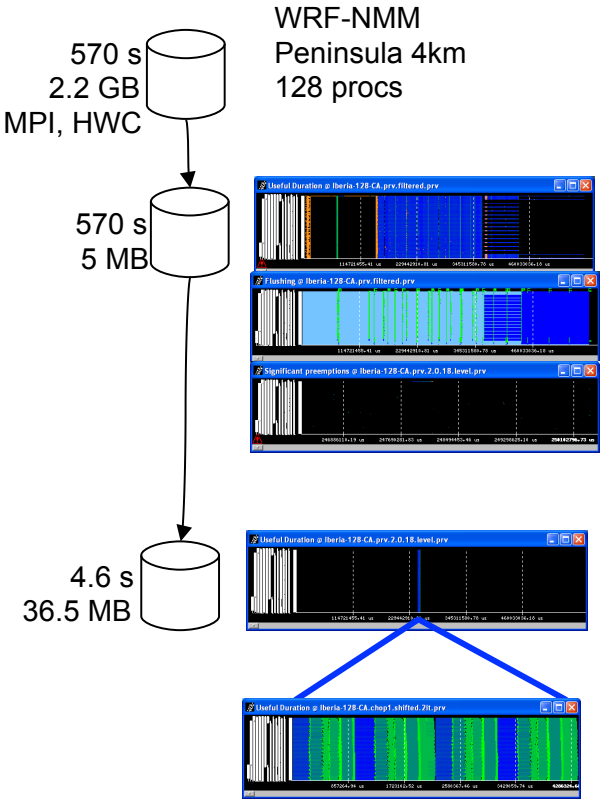






Trace manipulation

- Data handling/summarization capability
  - Filtering
    - Subset of records in original trace
    - By duration, type, value,…
    - Filtered trace IS a paraver trace and can be analysed with the same cfgs (as long as needed data kept)
  - Cutting
    - All records in a given time interval
    - Only some processes
  - Software counters
    - Summarized values computed from those in the original trace emitted as new even types
    - #MPI calls, total hardware count,…



www.bsc.es

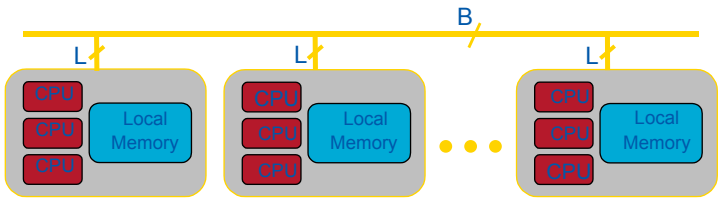
Barcelona Supercomputing Center

Centro Nacional de Supercomputación

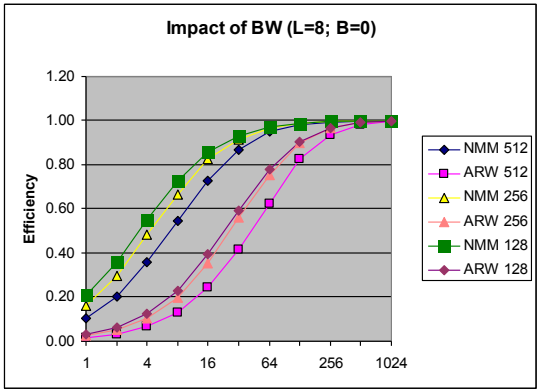
Dimemas

Dimemas: Coarse grain, Trace driven simulation

- Simulation: Highly non linear model
  - MPI protocols, resource contention…



- Parametric sweeps
  - On abstract architectures
  - On application computational regions
- What if analysis
  - Ideal machine (instantaneous network)
  - Estimating impact of ports to MPI+OpenMP/CUDA/…
  - Should I use asynchronous communications?
  - Are all parts equally sensitive to network?
- MPI sanity check
  - Modeling nominal

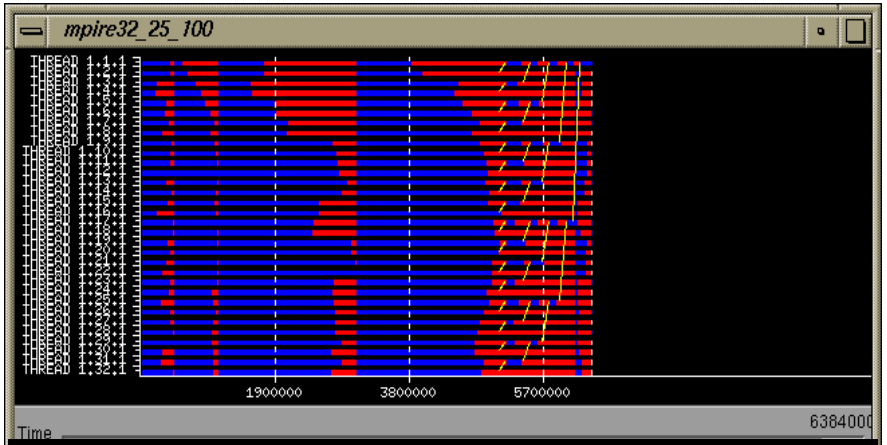


- Paraver – Dimemas tandem
  - Analysis and prediction
  - What-if from selected time window

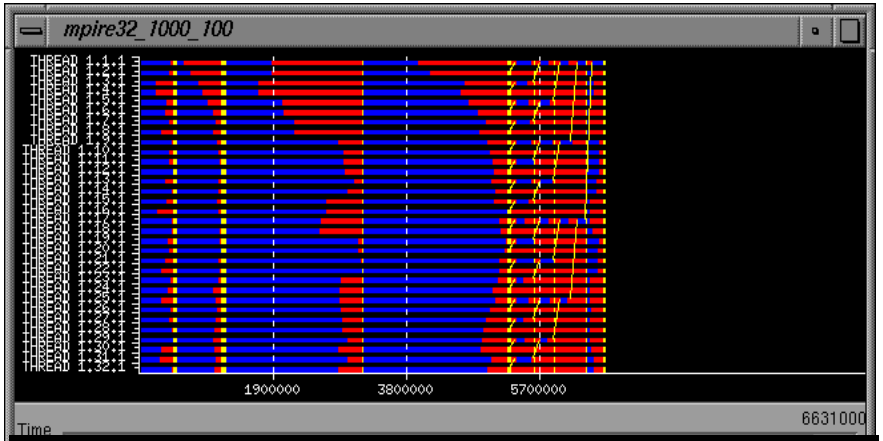
Detailed feedback on simulation (trace)

Network sensitivity

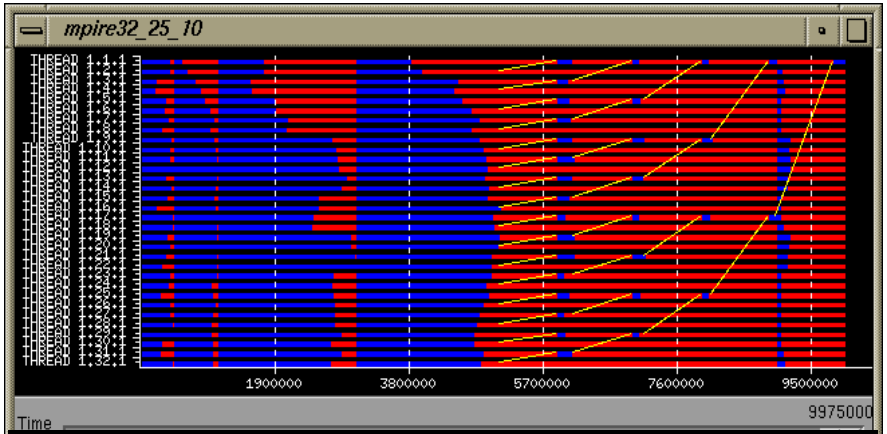
- MPIRE 32 tasks, no network contention



L = 5µs – BW = 1 GB/s



L = 1000µs – BW = 1 GB/s

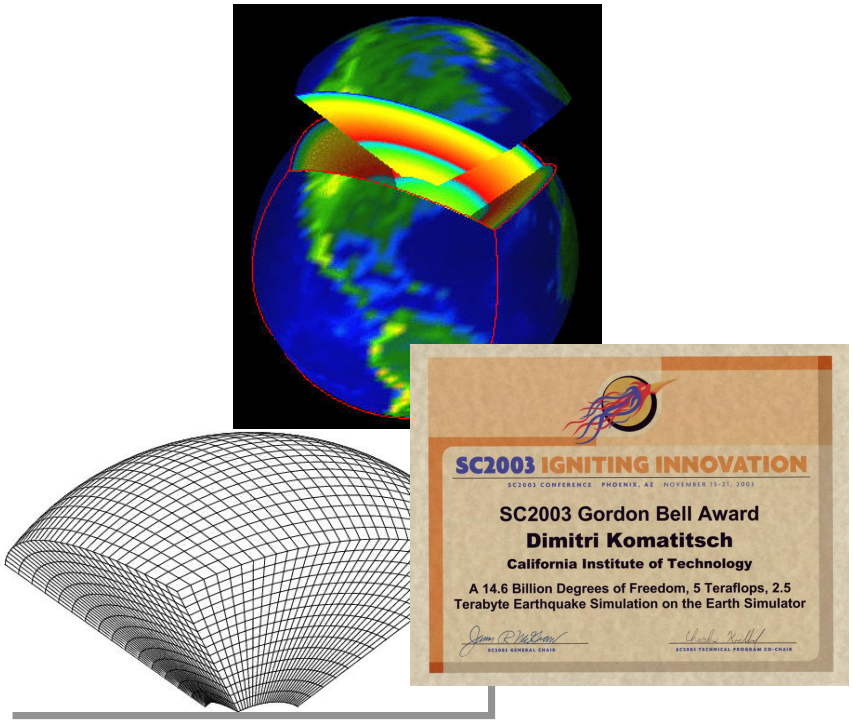


L = 5µs – BW = 100MB/s

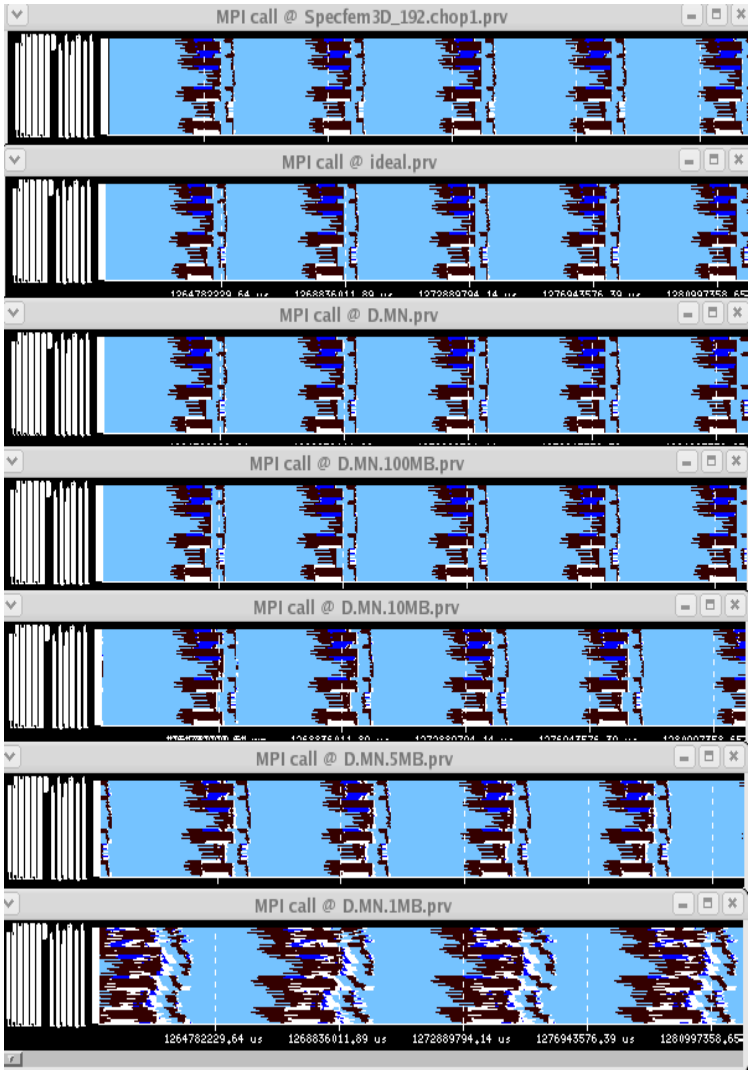
All windows same scale

Would I will benefit from asynchronous communications?

- SPECFEM3D



Courtesy Dimitri Komatitsch



Ideal machine

- The impossible machine: BW = ∞, L = 0
- Actually describes/characterizes Intrinsic application behavior
  - Load balance problems?
  - Dependence problems?

GADGET @ Nehalem cluster
256 processes

Allgather + sendrecv

allreduce

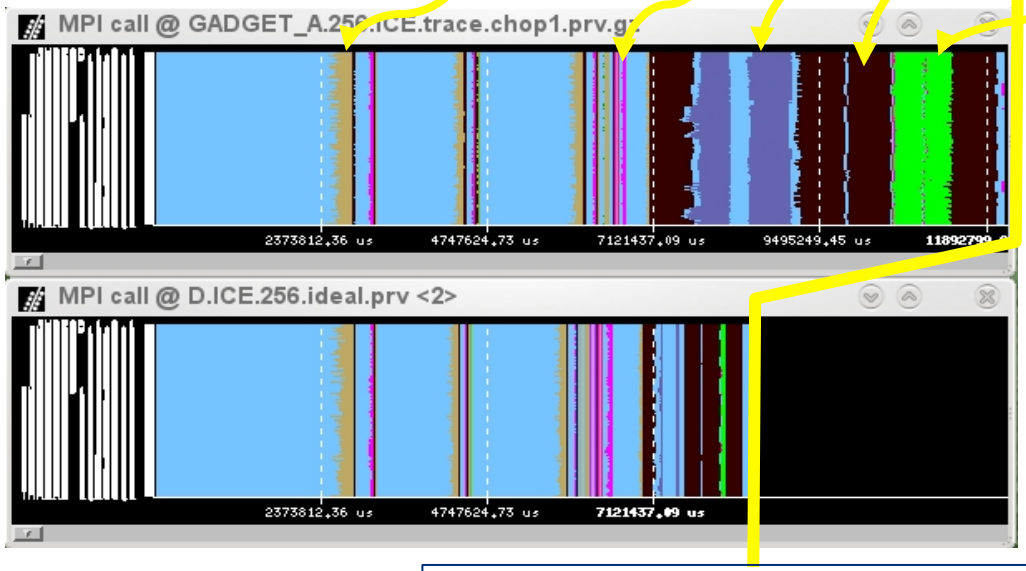
alltoall

sendrec

waitall

Real run

Ideal network

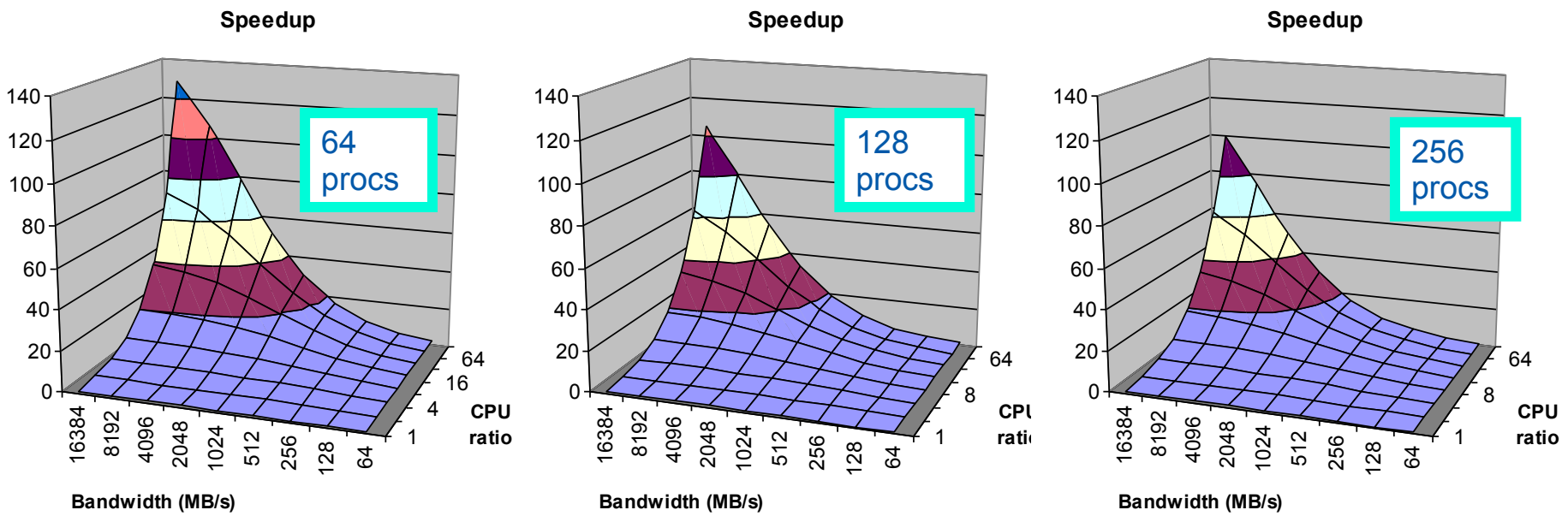


Impact on practical machines?

Impact of architectural parameters

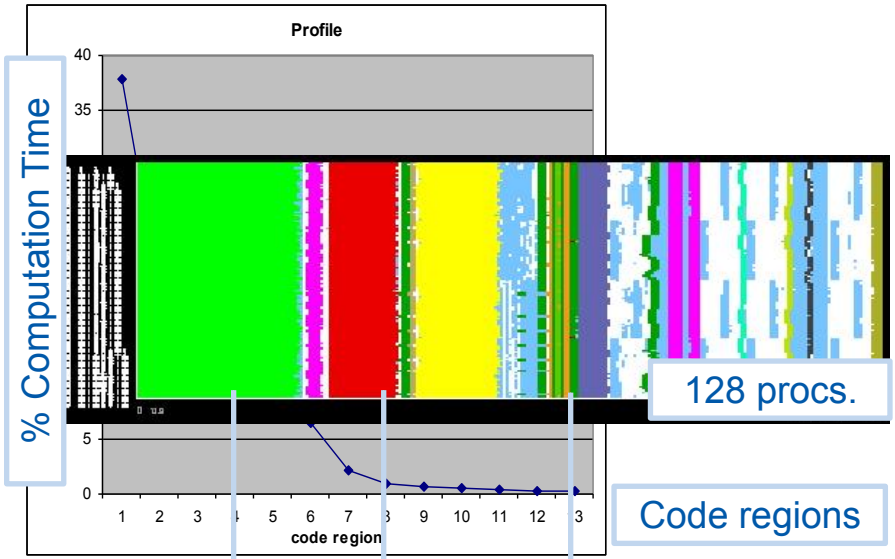
- **Ideal speeding up ALL** the computation bursts by the CPUratio factor
  - The more processes the less speedup (higher impact of bandwidth limitations) !!

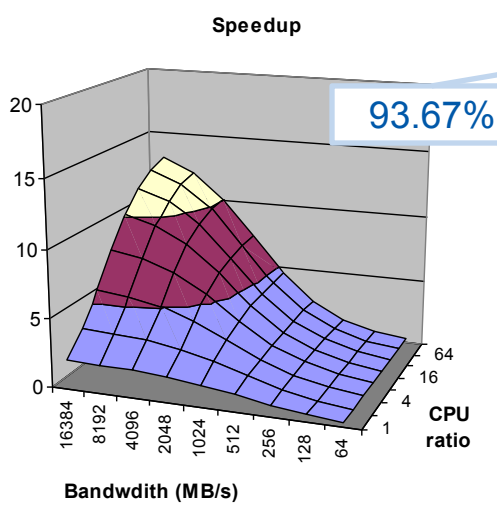
GADGET

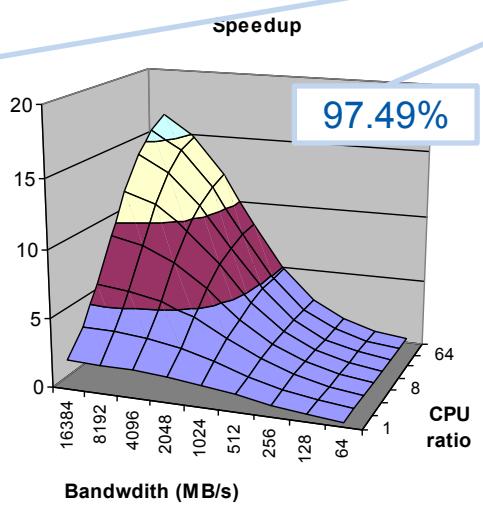


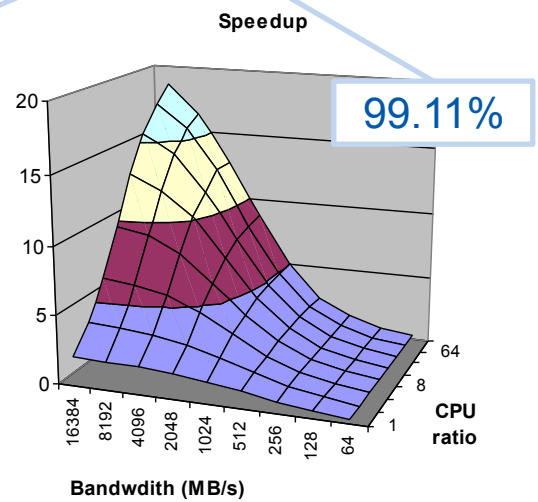
Hybrid parallelization

- Hybrid/accelerator parallelization
  - Speed-up SELECTED regions by the CPUratio factor









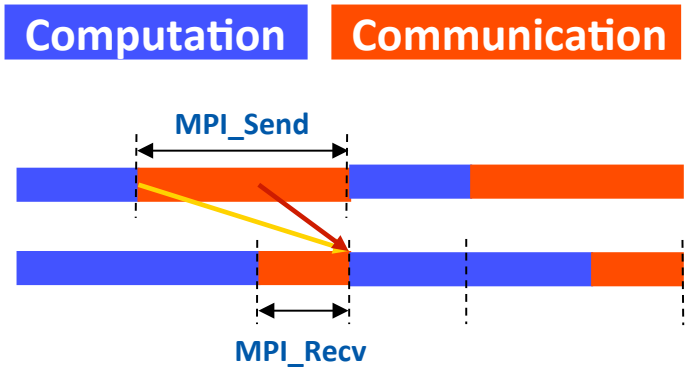
(Previous slide: speedups **up to 100x**)

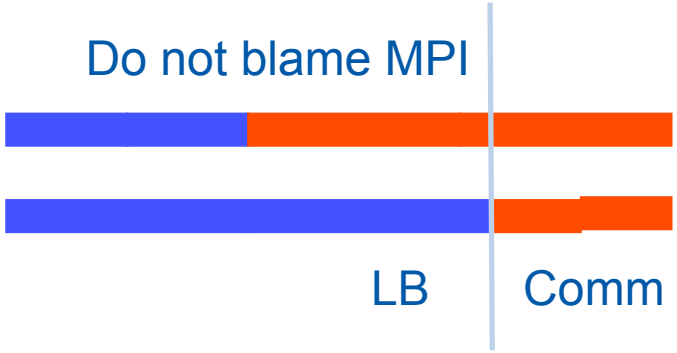
www.bsc.es

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Models and Extrapolation

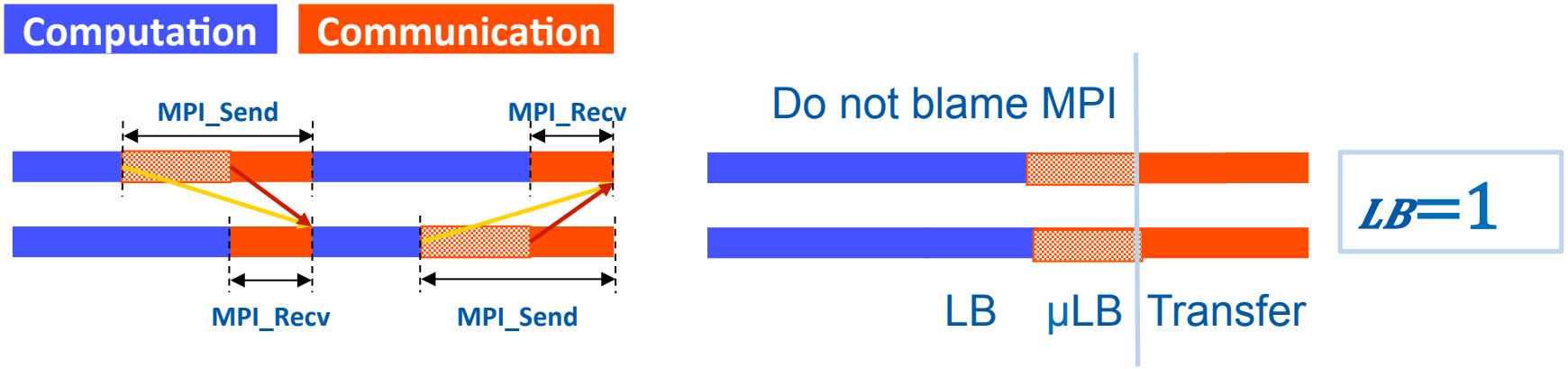
Parallel efficiency model



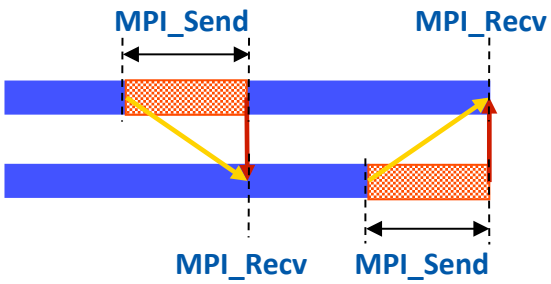


- Parallel efficiency = LB eff * Comm eff

Parallel efficiency refinement: LB * µLB * Transfer



- Serializations / dependences (µLB)
- Dimemas ideal network → Transfer (efficiency) = 1

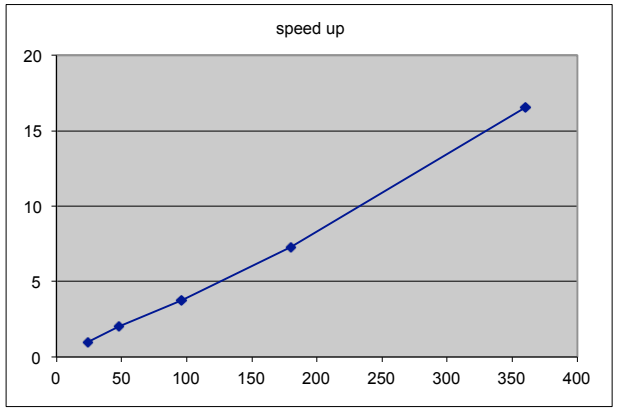


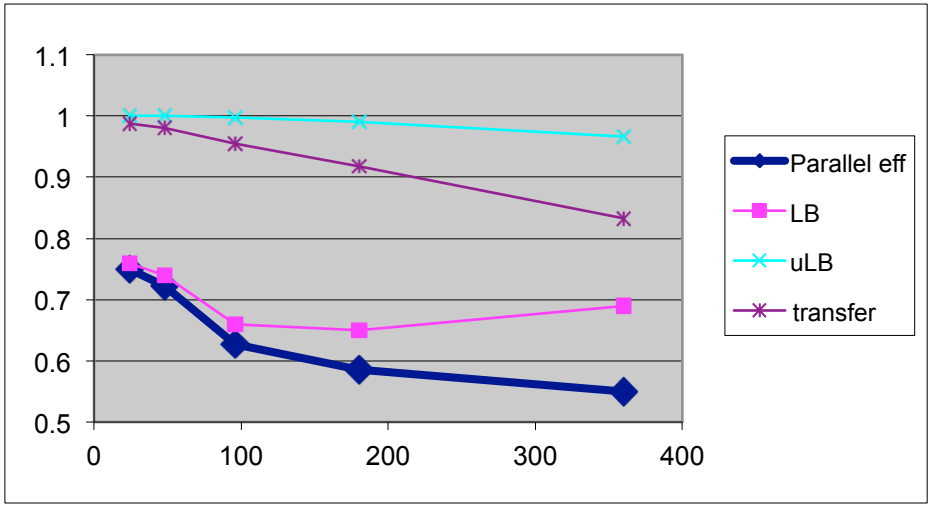
Why scaling?

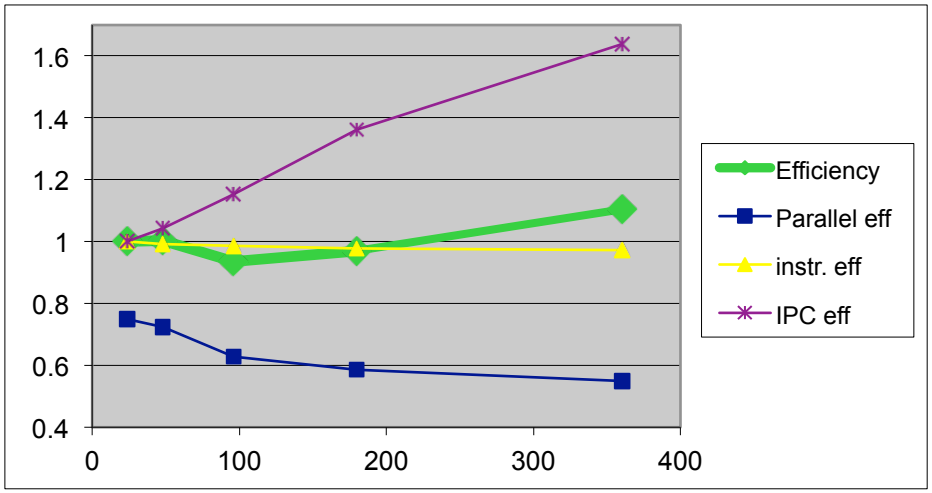
$$\eta_{\parallel} = LB * Ser * Trf$$

CG-POP mpi2s1D - 180x120

Good scalability !!
Should we be happy?



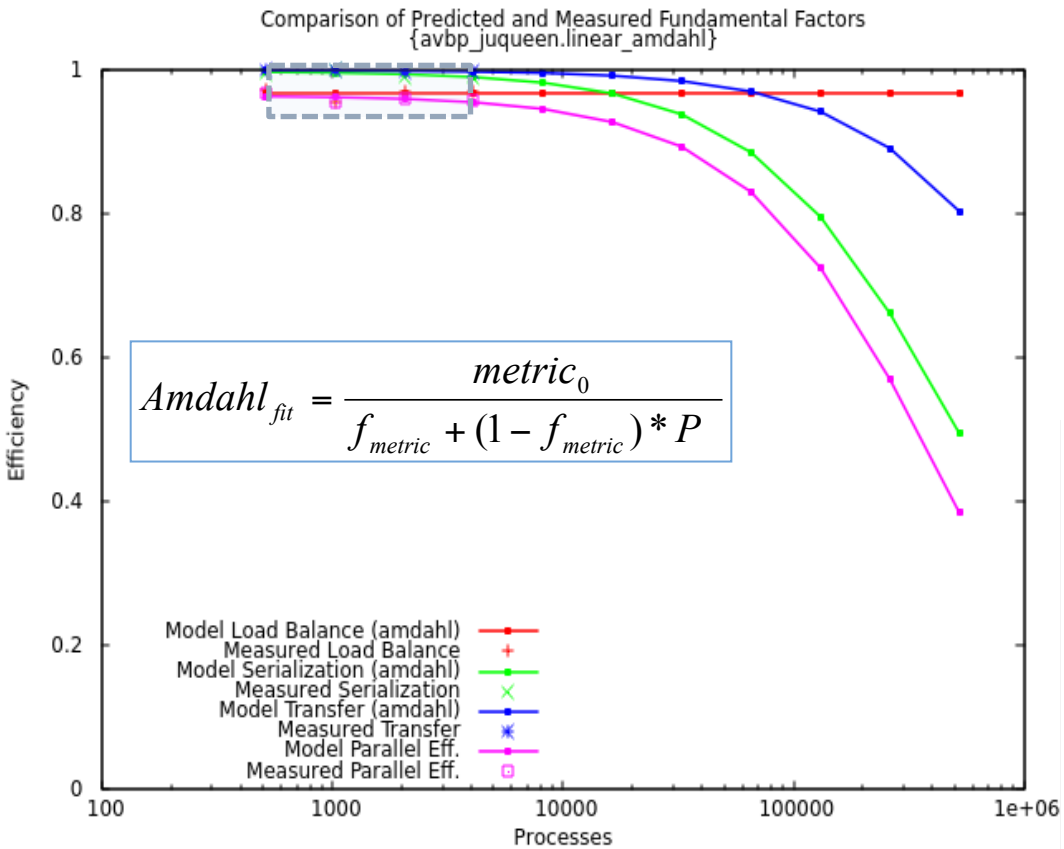




$$\eta = \eta_{\parallel} * \eta_{instr} * \eta_{IPC}$$

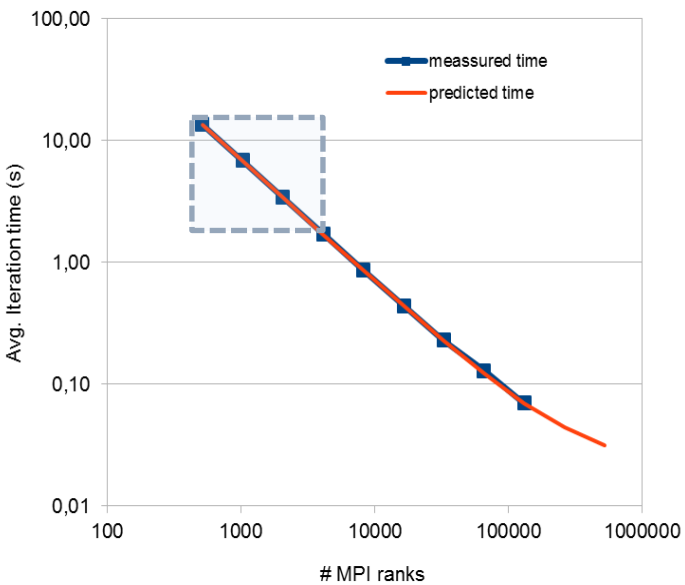
AVBP (strong scale)

- Input: runs 512 – 4096



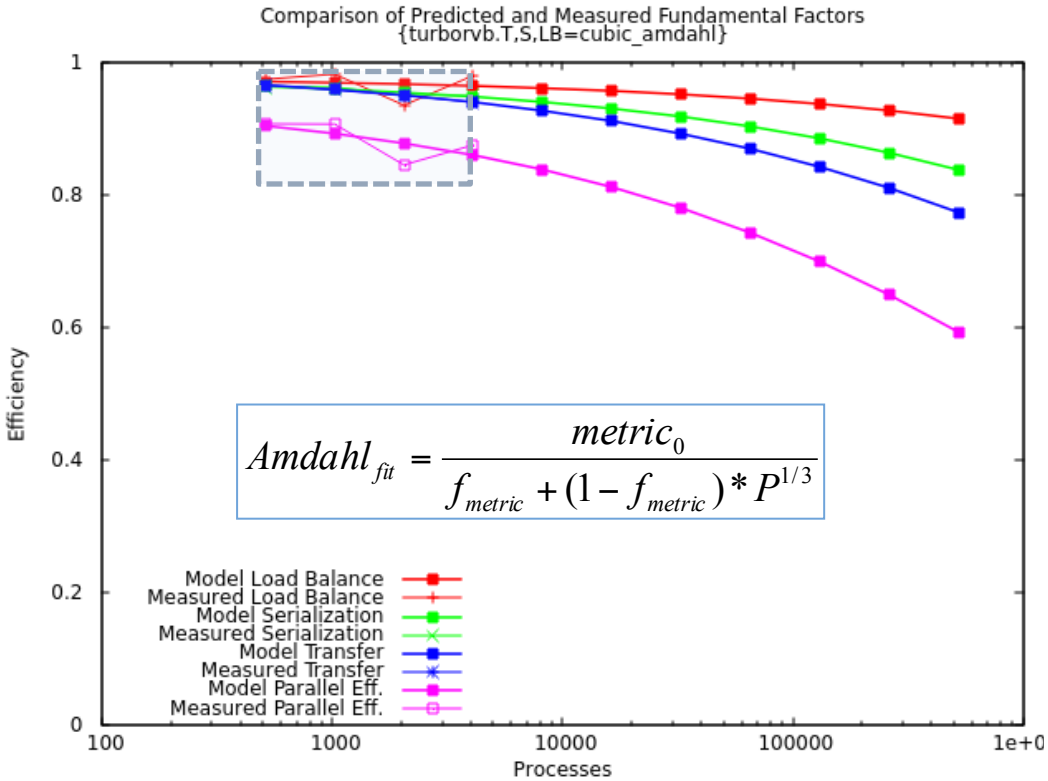
$$Amdahl_{fit} = \frac{metric_0}{f_{metric} + (1 - f_{metric}) * P}$$

Strong scale → small computations betweeen MPI calls become more and more important



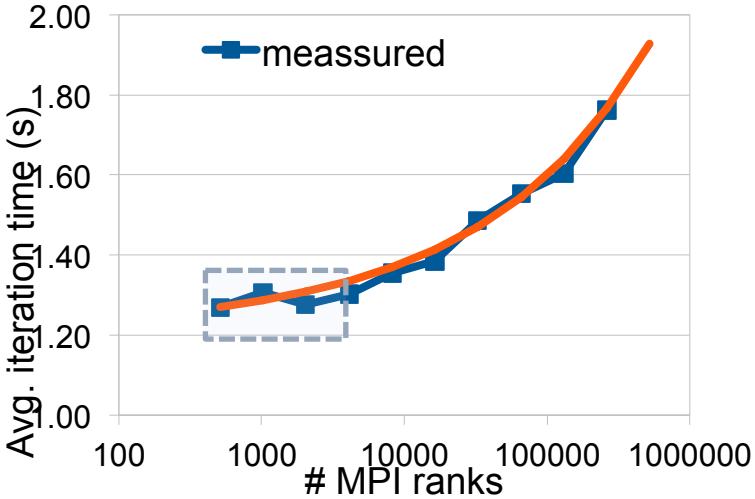
TURBORVB(strong scale)

- Input: runs 512 – 4096



$$Amdahl_{fit} = \frac{metric_0}{f_{metric} + (1 - f_{metric}) * P^{1/3}}$$

Network contention can be reduced balancing / limiting the random selection within a node

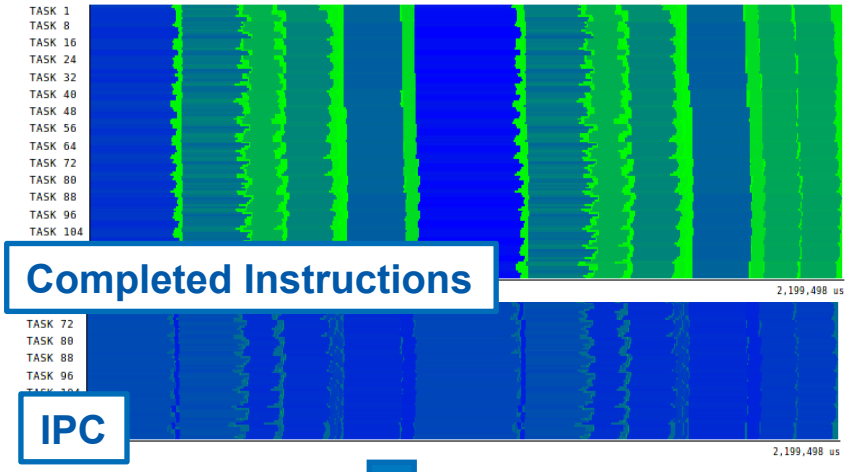


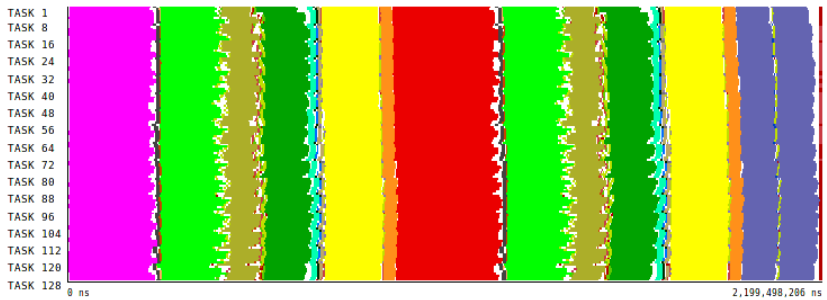
www.bsc.es

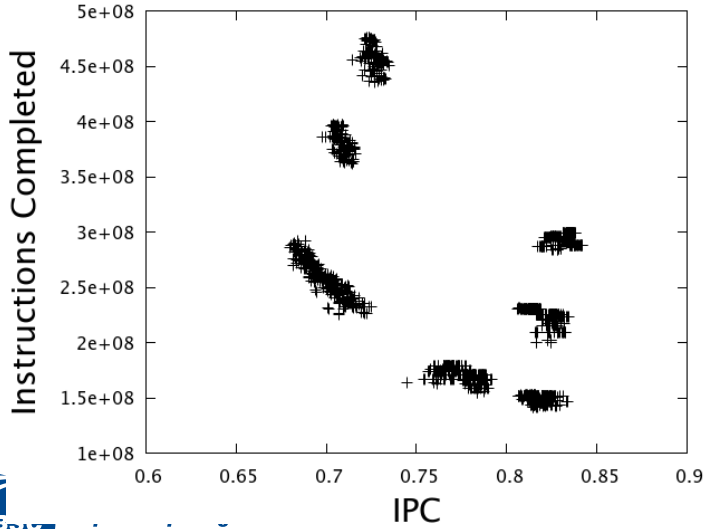
Barcelona Supercomputing Center
Centro Nacional de Supercomputación

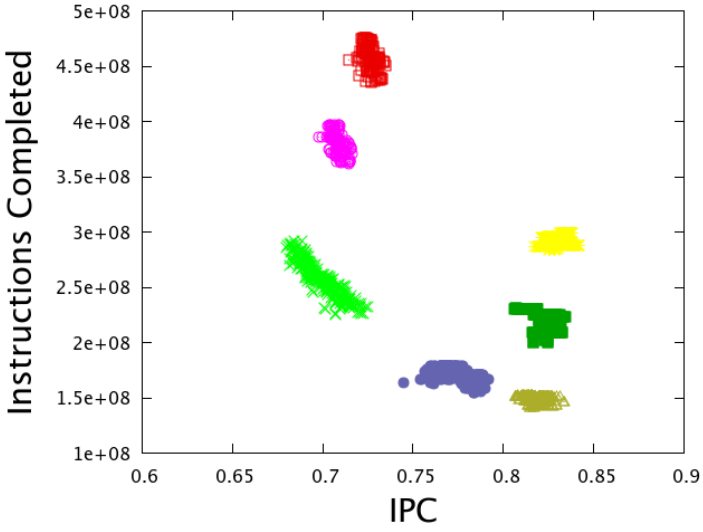
Clustering

Using Clustering to identify structure







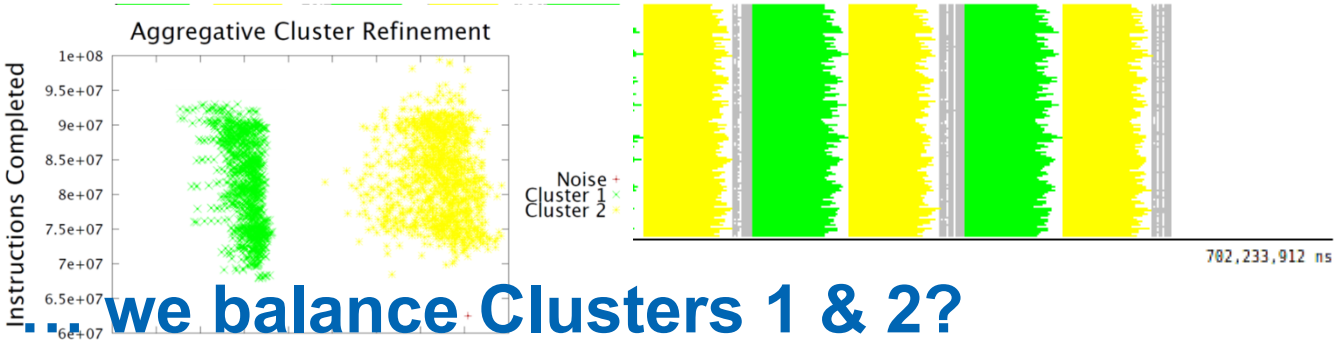


What should I improve?

What if ….

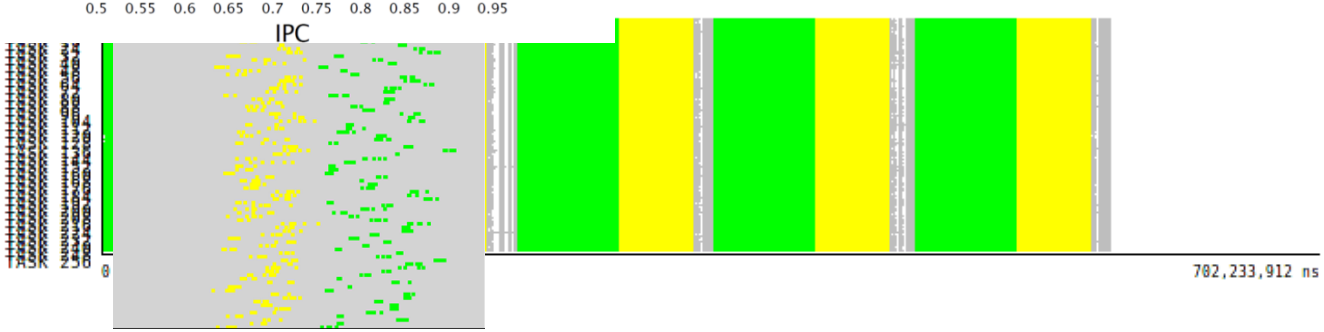
PEPC

… we increase the IPC of Cluster1?



13% gain

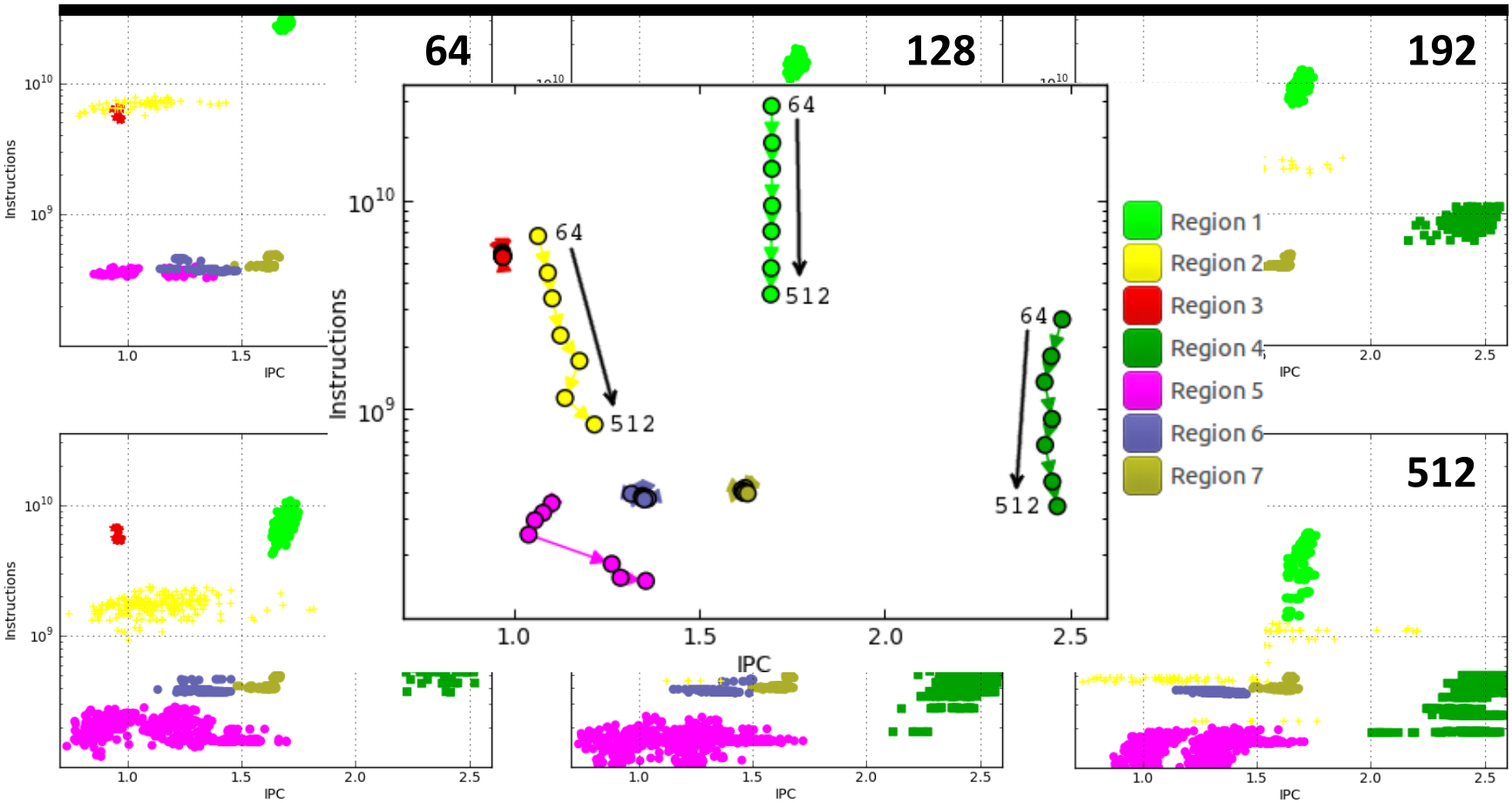
… we balance Clusters 1 & 2?



19% gain

Tracking scability through clustering

- OpenMX (strong scale from 64 to 512 tasks)



www.bsc.es

Barcelona Supercomputing Center

Centro Nacional de Supercomputación

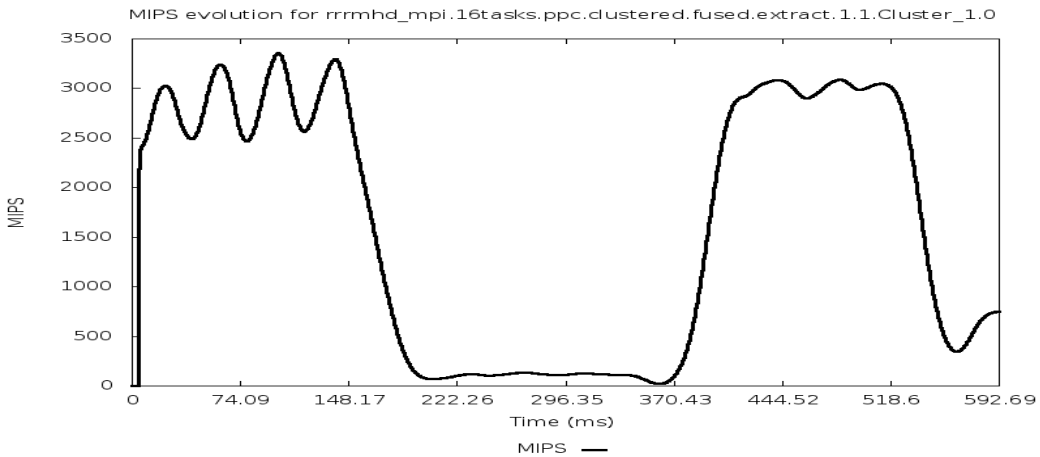
Folding

Folding: Detailed metrics evolution

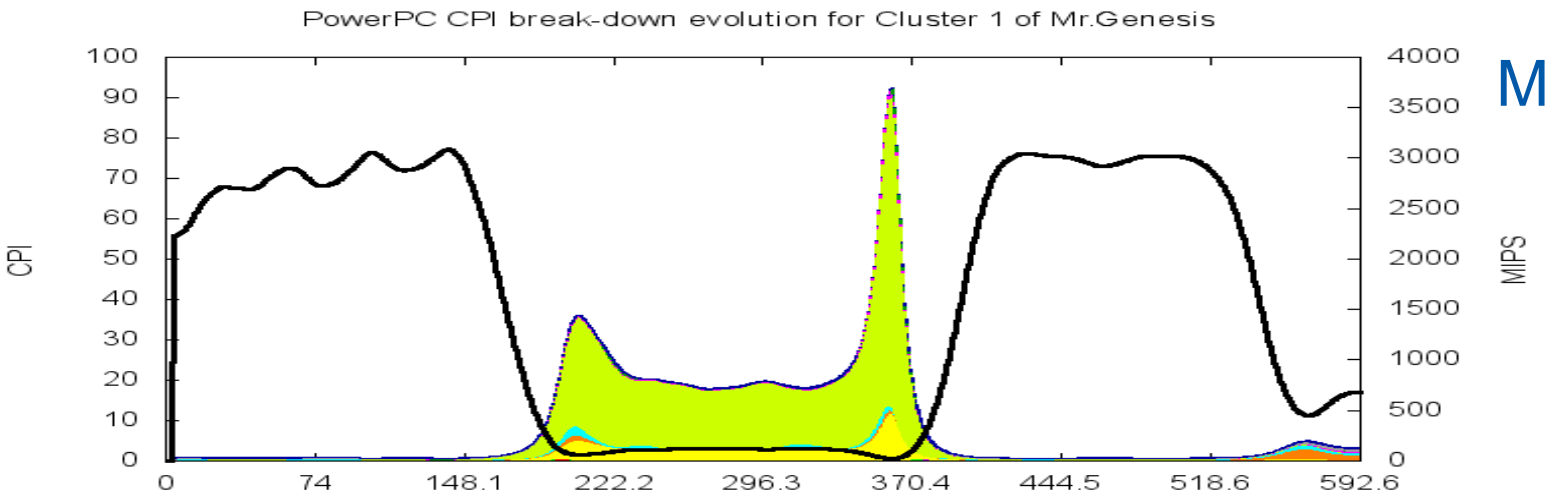
- Performance of a sequential region = 2000 MIPS

Is it good enough?

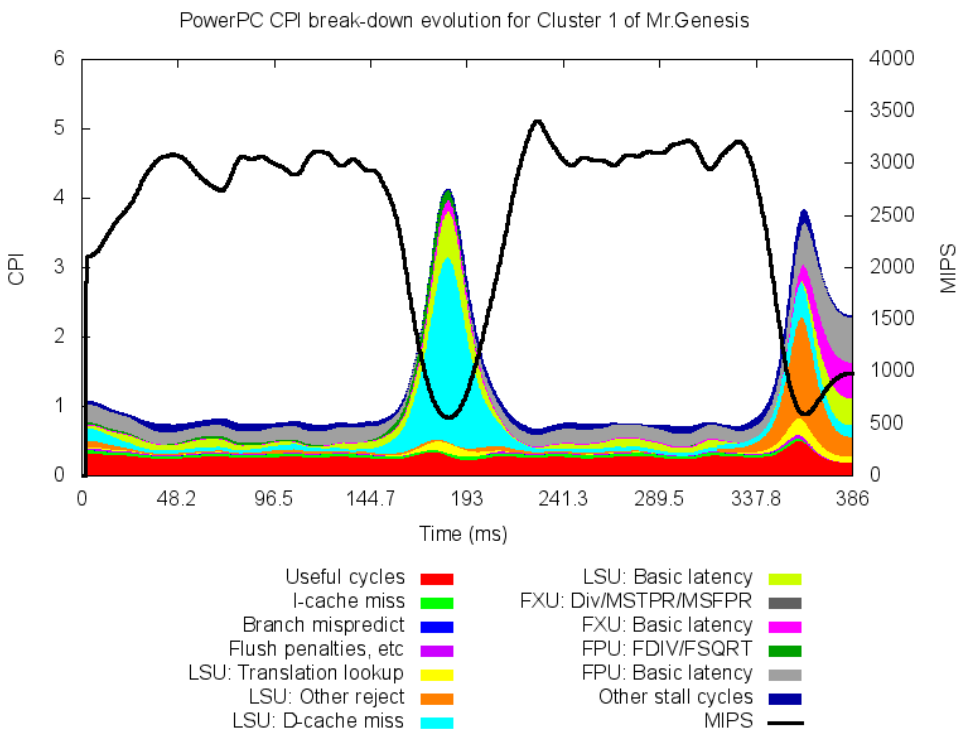
Is it easy to improve?



Folding: Instantaneous CPI stack



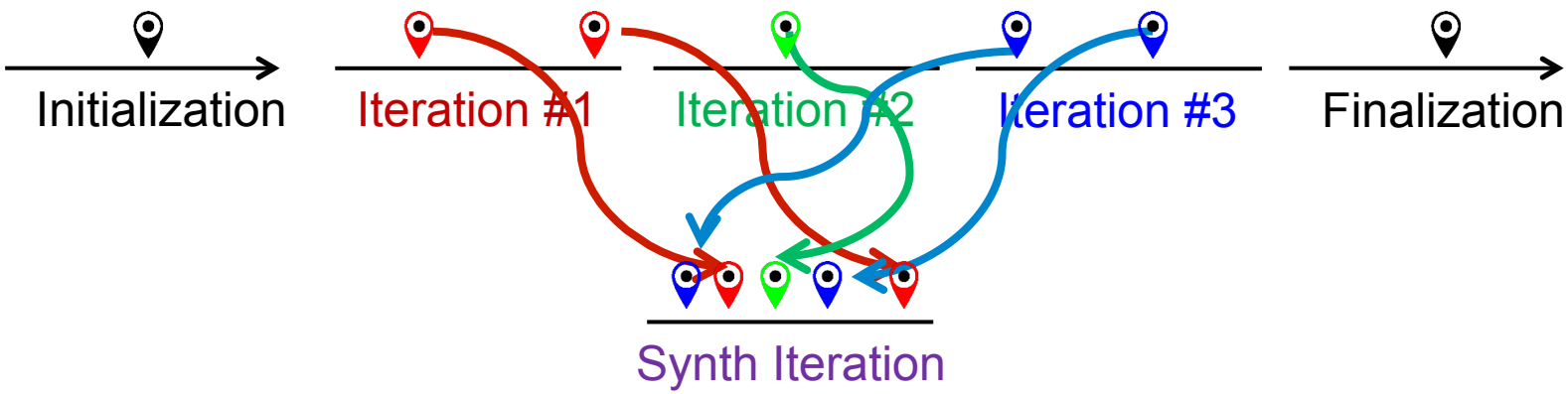
MRGENESIS



- Trivial fix.(loop interchange)
- Easy to locate?
- Next step?
- Availability of CPI stack models for production processors?
  - Provided by manufacturers?

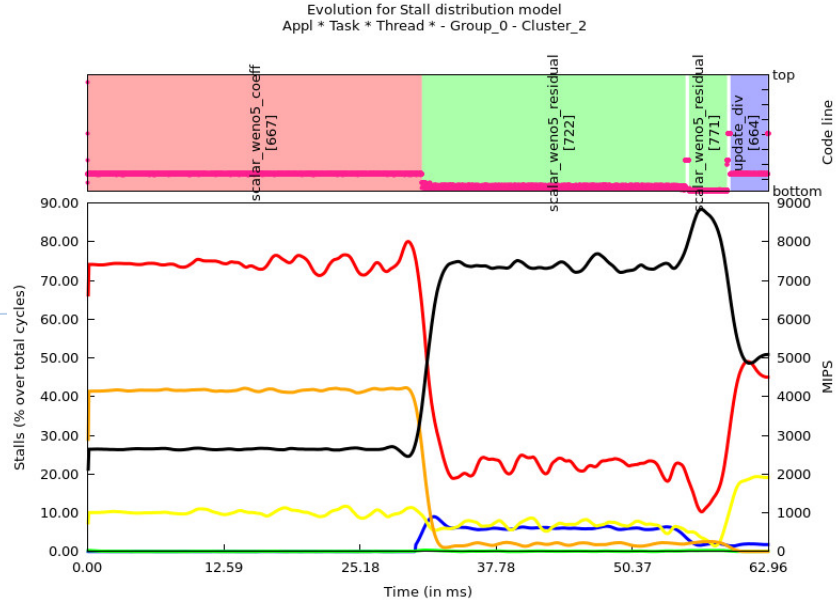
Folding

- Instantaneous metrics with minimum overhead
  - Combine instrumentation and sampling
    - Instrumentation delimits regions (routines, loops, …)
    - Sampling exposes progression within a region
  - Captures performance counters and call-stack references

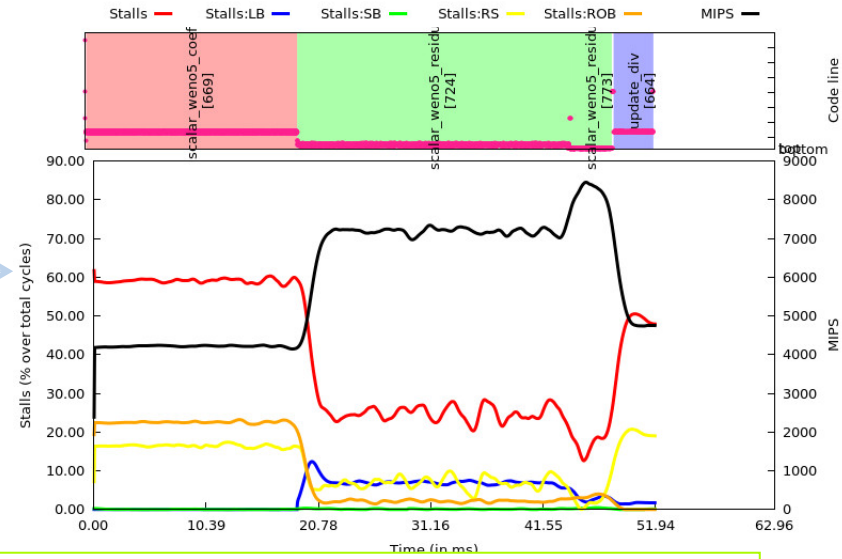


“Blind” optimization

- From folded samples of a few levels to timeline structure of “relevant” routines



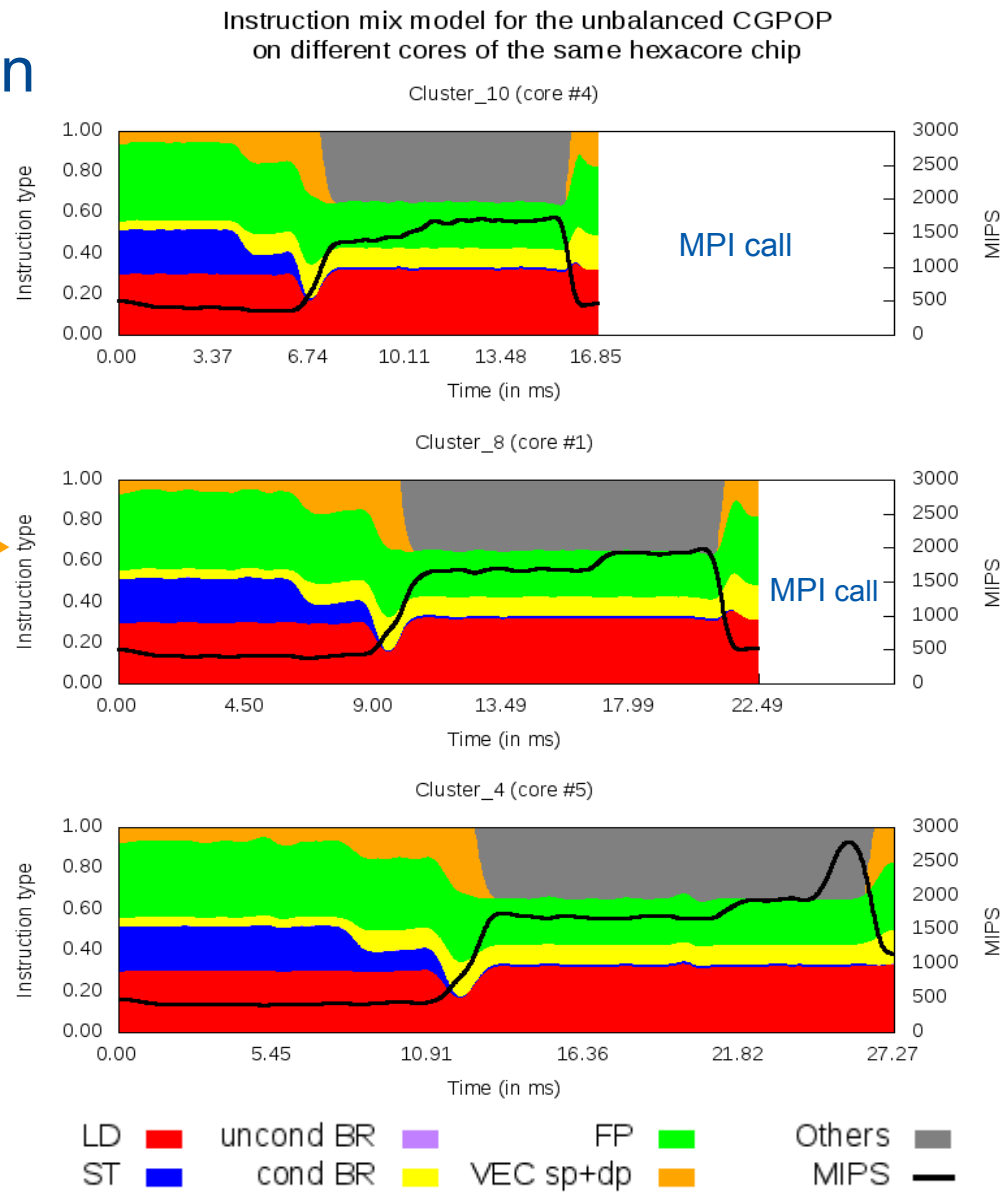
Recommendation without access to source code



Harald Servat et al. “Bio-inspired call-stack reconstruction for performance analysis” Submitted, 2015

CG-POP multicore MN3 study

- Unbalanced MPI application
  - Same code
  - Different duration
  - Different performance

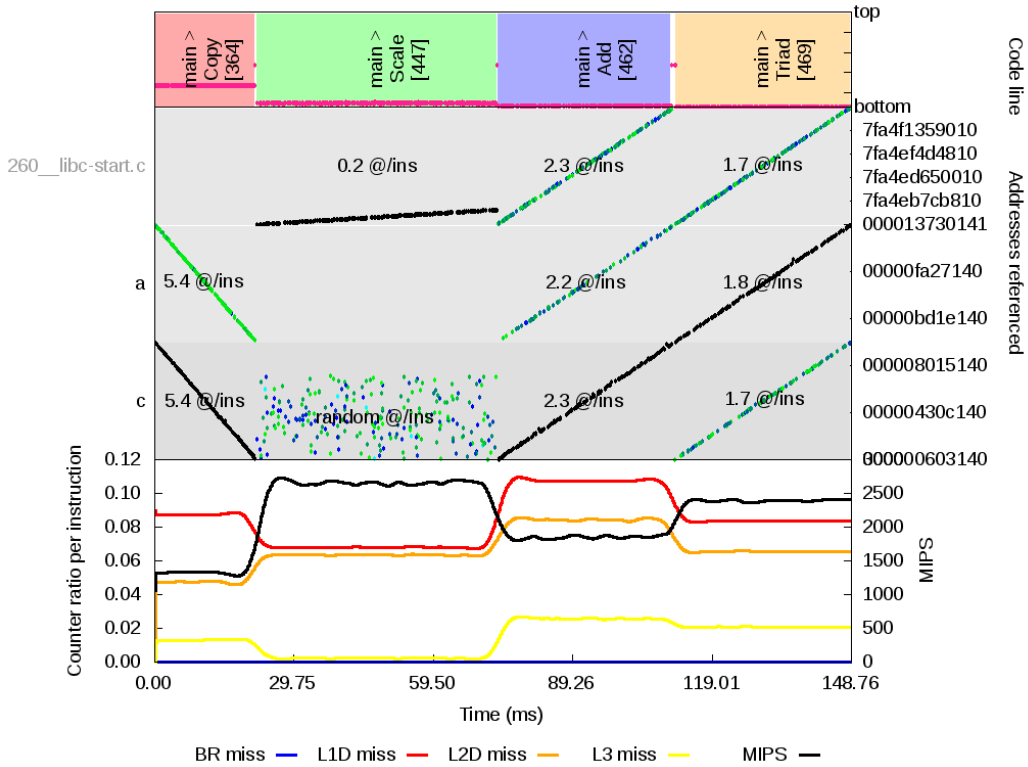


Address space references

- Based on PEBS events
  - Loads: address, cost in cycles, level providing the data
  - Stores: only address

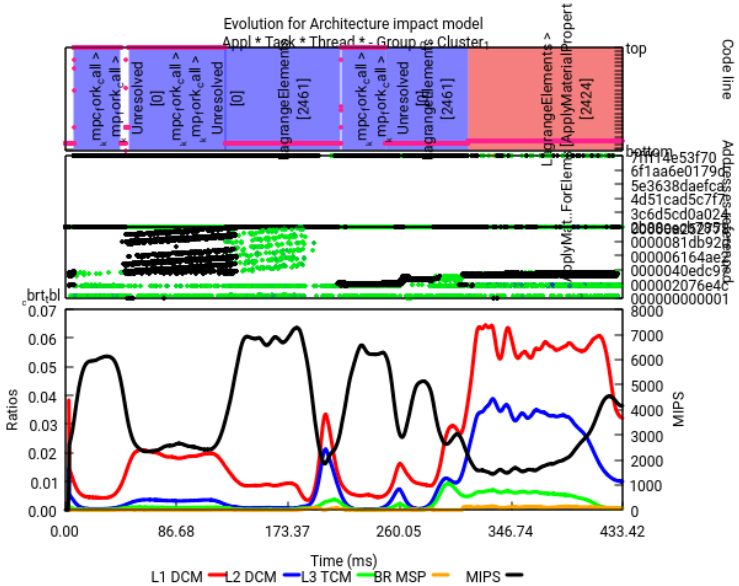
Modified Stream

```
for (i = 0; i < NITERS; i++) {
    Extrae_event (BEGIN);
    for (j = 0; j < N; j++)
        c[j] = a[j];
    for (j = 0; j < N/8; j++)
        b[j] = s*c[random()%j];
    for (j = 0; j < N; j++)
        c[j] = a[j]+b[j];
    for (j = 0; j < N; j++)
        a[j] = b[j]+s*c[j];
   Extrae_event (END);
}
```

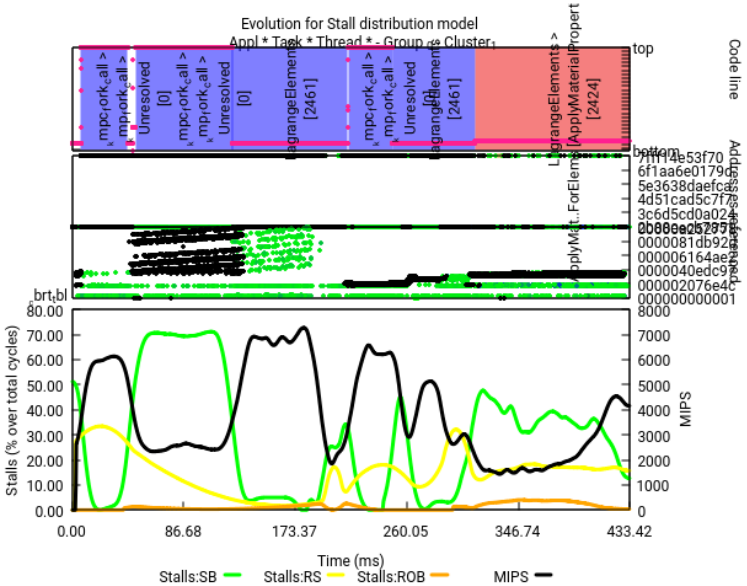


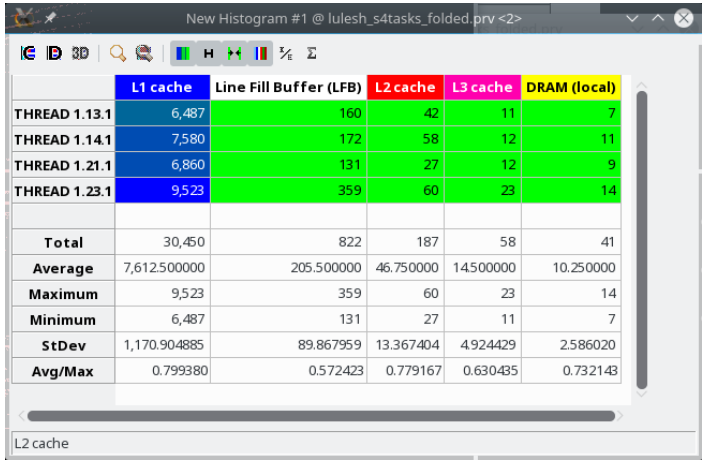
Lulesh memory access

Architecture impact

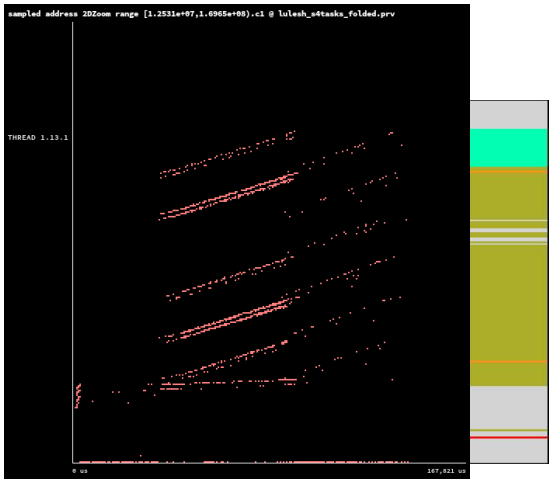


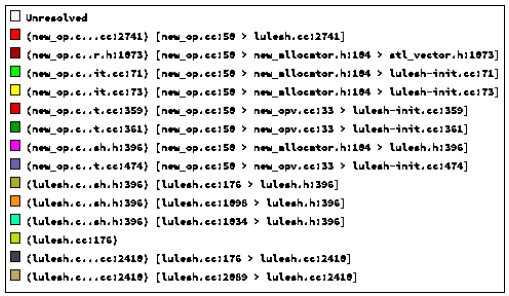
Stalls distribution





	L1 cache	Line Fill Buffer (LFB)	L2 cache	L3 cache	DRAM (local)
THREAD 1.13.1	6,487	160	42	11	7
THREAD 1.14.1	7,580	172	58	12	11
THREAD 1.21.1	6,860	131	27	12	9
THREAD 1.23.1	9,523	359	60	23	14
Total	30,450	822	187	58	41
Average	7,612.500000	205.500000	46.750000	14.500000	10.250000
Maximum	9,523	359	60	23	14
Minimum	6,487	131	27	11	7
StDev	1,170.904885	89.867959	13.367404	4.924429	2.586020
Avg/Max	0.799380	0.572423	0.779167	0.630435	0.732143





www.bsc.es

Barcelona
Supercomputing
Center
Centro Nacional de Supercomputación

Methodology

Performance analysis tools objective

Help generate hypotheses

Help validate hypotheses

Qualitatively

Quantitatively

First steps

- Parallel efficiency – percentage of time invested on computation
  - Identify sources for “inefficiency”:
    - load balance
    - Communication /synchronization
- Serial efficiency – how far from peak performance?
  - IPC, correlate with other counters
- Scalability – code replication?
  - Total #instructions
- Behavioral structure? Variability?

BSC Tools web site

- www.bsc.es/paraver
  - downloads
    - Sources / Binaries
    - Linux / windows / MAC
  - documentation
    - Training guides
    - Tutorial slides

- Getting started
  - Start wxparaver
  - Help → tutorials and follow instructions
  - Follow training guides
    - Paraver introduction (MPI): Navigation and basic understanding of Paraver operation

Thanks!

- Use your brain, use visual tools :)
- Look at your codes!

www.bsc.es/paraver

www.bsc.es

Barcelona Supercomputing Center

Centro Nacional de Supercomputación

Demo

Some examples of efficiencies

Code	Parallel efficiency	Communication efficiency	Load Balance efficiency
Gromacs@mt	66.77	75.68	88.22
BigDFT@altamira	59.64	78.97	75.52
CG-POP@mt	80.98	98.92	81.86
ntchem_mini@pi	92.56	94.94	97.49
nicam@pi	87.10	75.97	89.22
cp2k@jureca	75.34	81.07	92.93
lulesh@mn3	90.55	99.22	91.26
lulesh@leftraru	69.15	99.12	69.76
lulesh@uv2 (mpt)	70.55	96.56	73.06
lulesh@uv2 (impi)	85.65	95.09	90.07
lulesh@mt	83.68	95.48	87.64
lulesh@cori	90.92	98.59	92.20