

# Understanding applications using the BSC performance tools

---

Judit Gimenez (judit@bsc.es), Lau Mercadal

Barcelona Supercomputing Center

---

# Humans are visual creatures

---

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

PROCESS

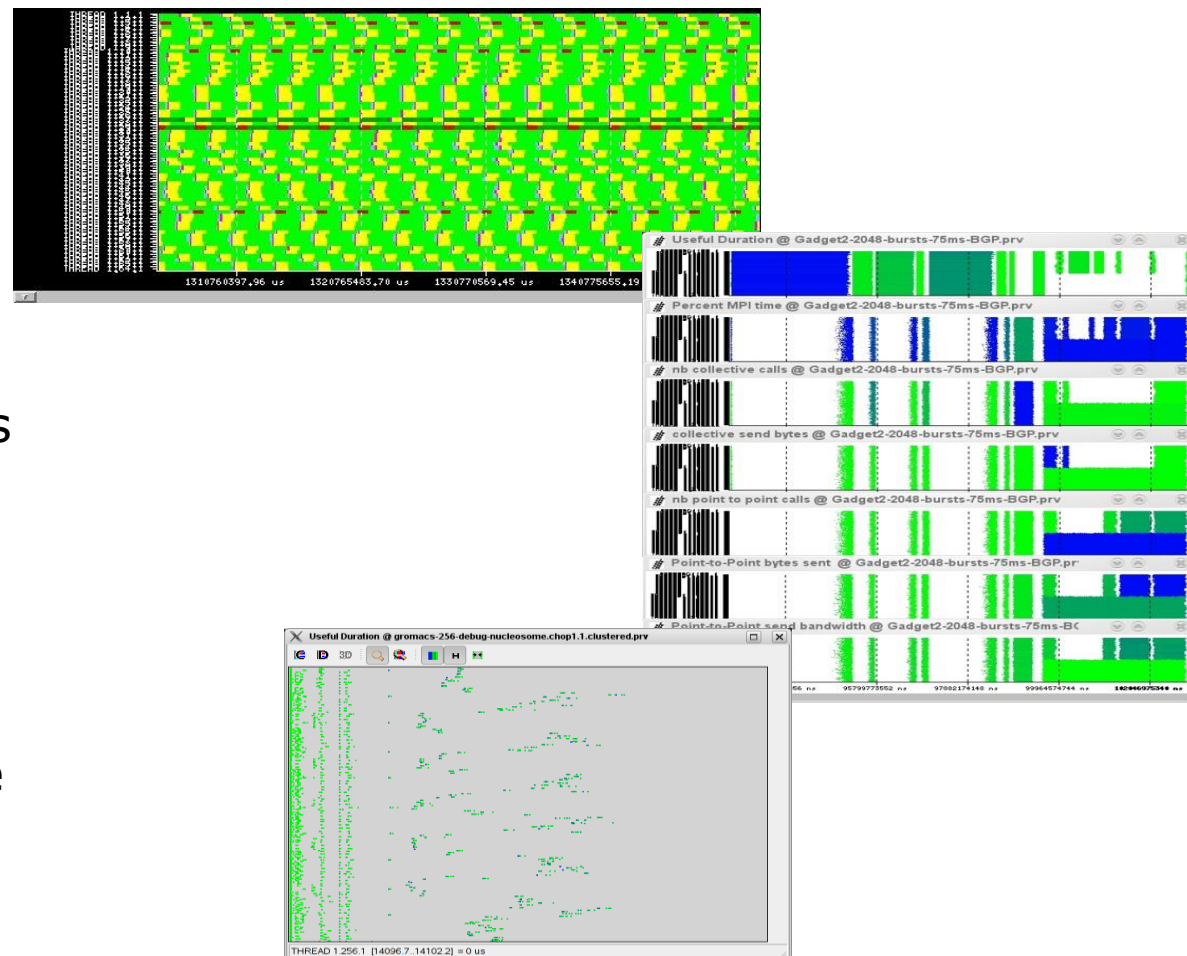
STORE

IDENTIFY



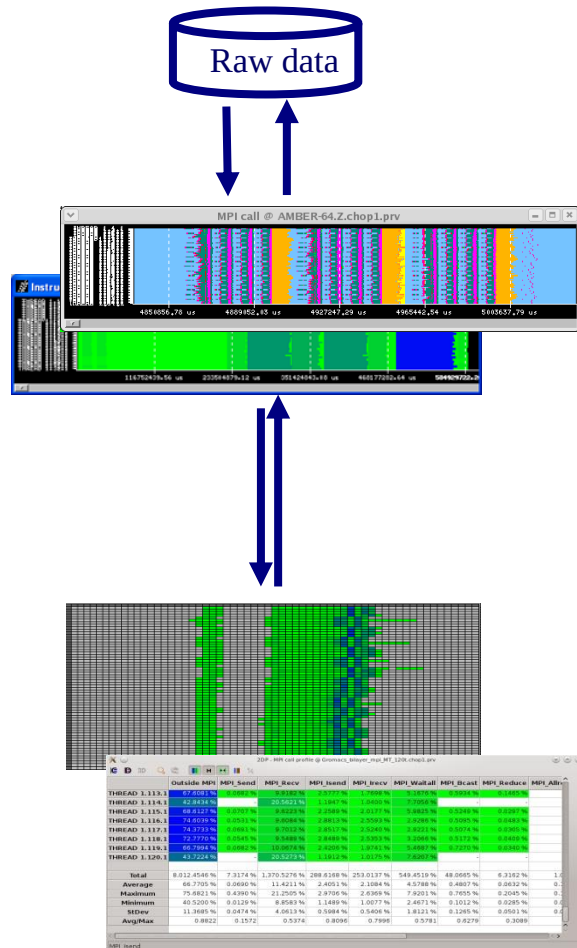
# Our Tools

- 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



# Paraver

# Paraver: Performance data browser



Trace visualization/analysis

+ trace manipulation

Timelines

2/3D tables  
(Statistics)

Goal = Flexibility

No semantics

Programmable

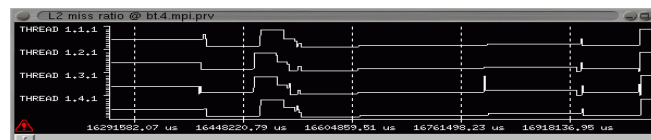
Comparative analyses

Multiple traces

Synchronize scales

# Timelines

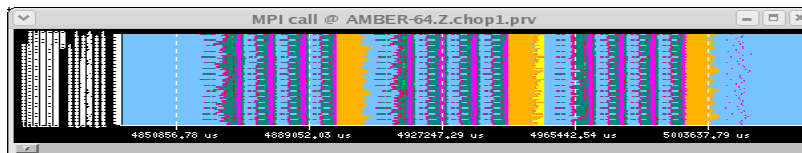
- Each window displays one view
  - Piecewise constant** function of time



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

- Types of functions

- Categorical
  - State, user function, outlined routine

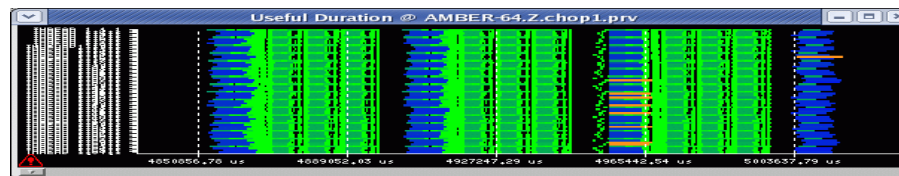


$$S_i \in [0, n] \subset N, \quad n <$$

- 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

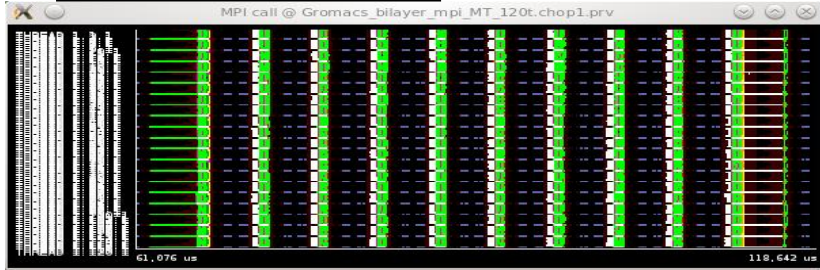


$$S_i \in R$$

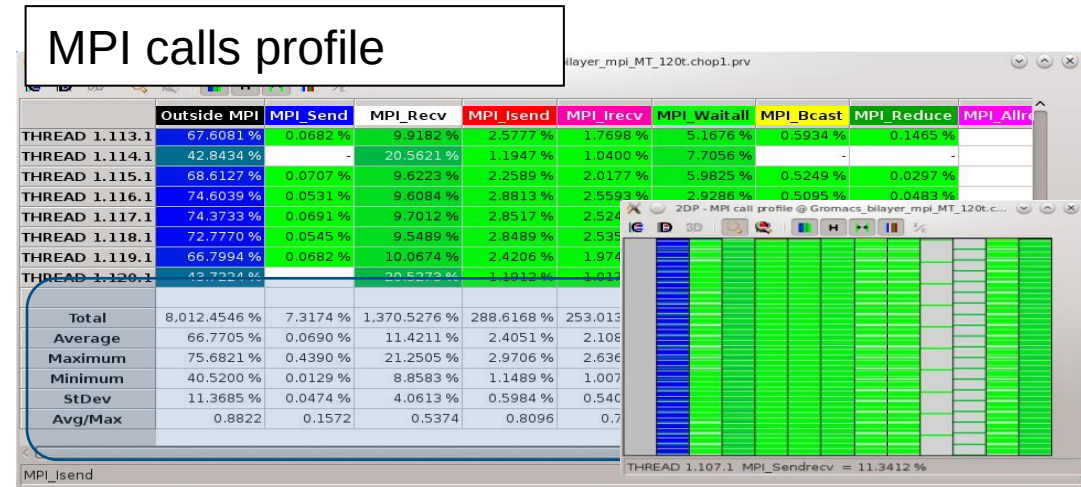
# Tables: Profiles, histograms, correlations

- From timelines to tables

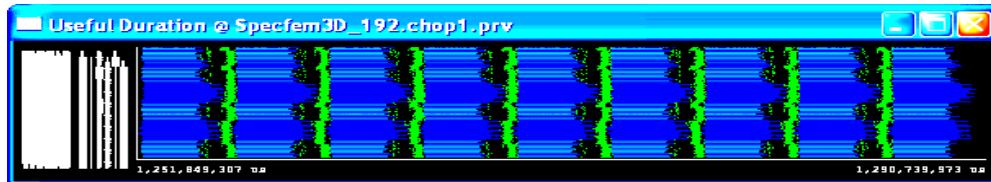
MPI calls



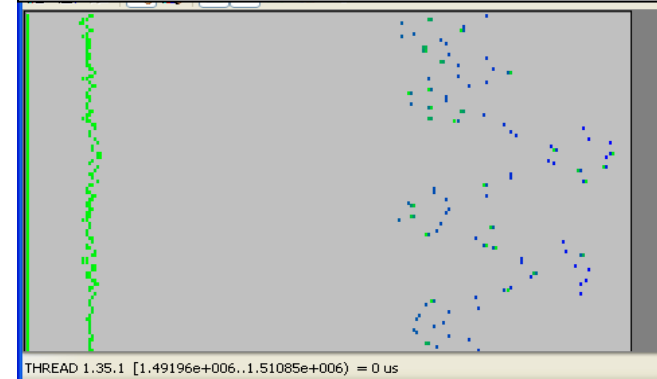
MPI calls profile



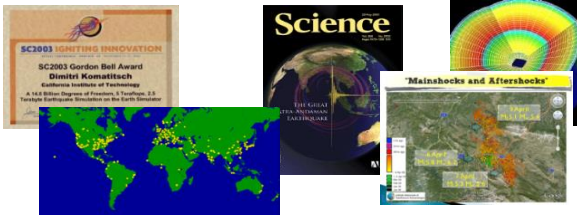
Useful Duration



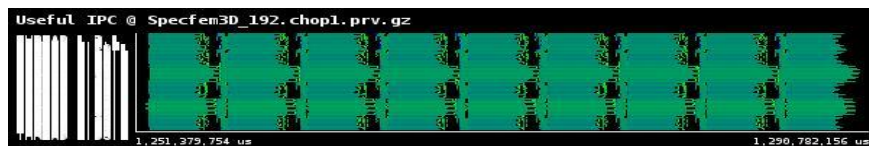
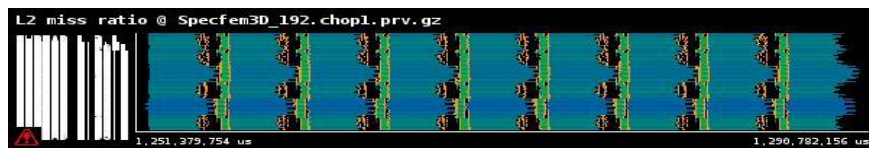
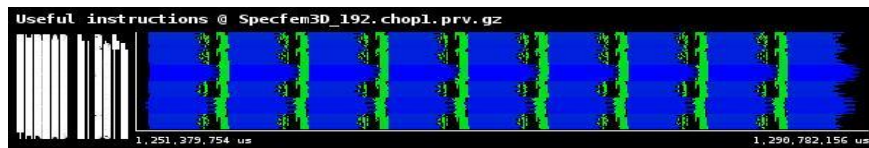
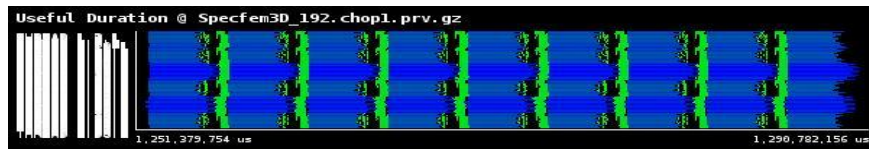
Histogram Useful Duration



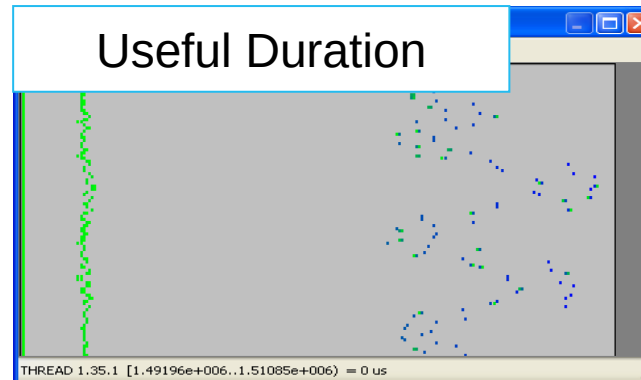
# Analyzing variability through histograms and timelines



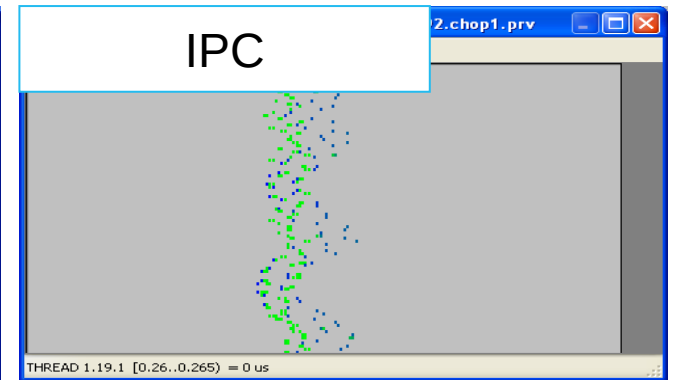
SPECFEM3D



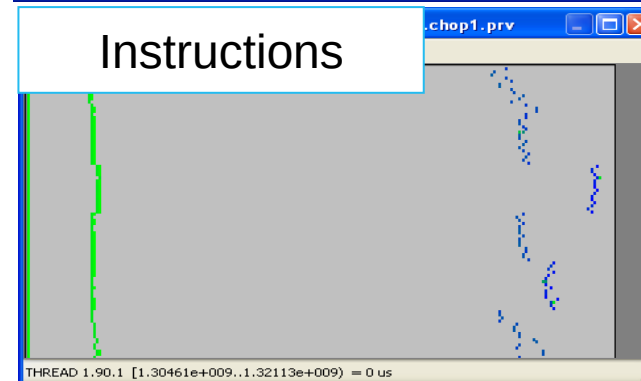
Useful Duration



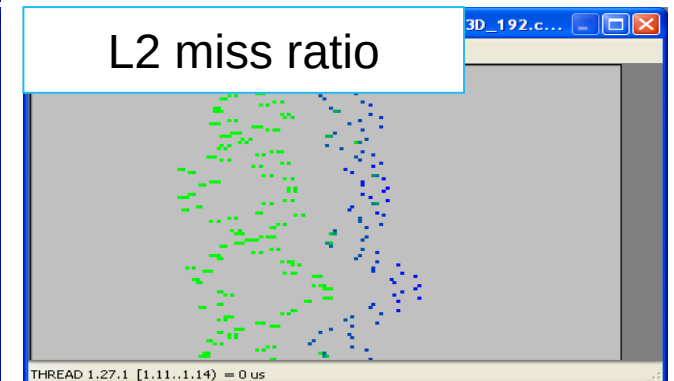
IPC



Instructions

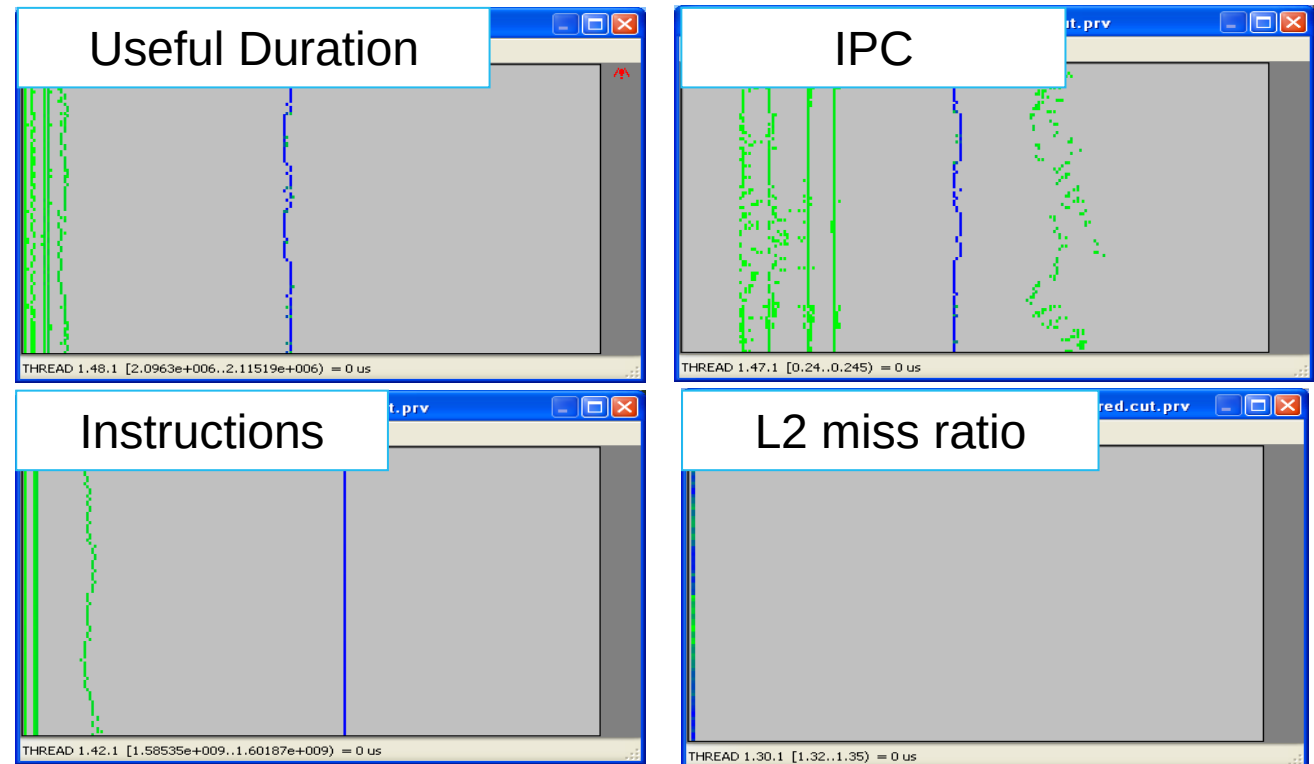


L2 miss ratio



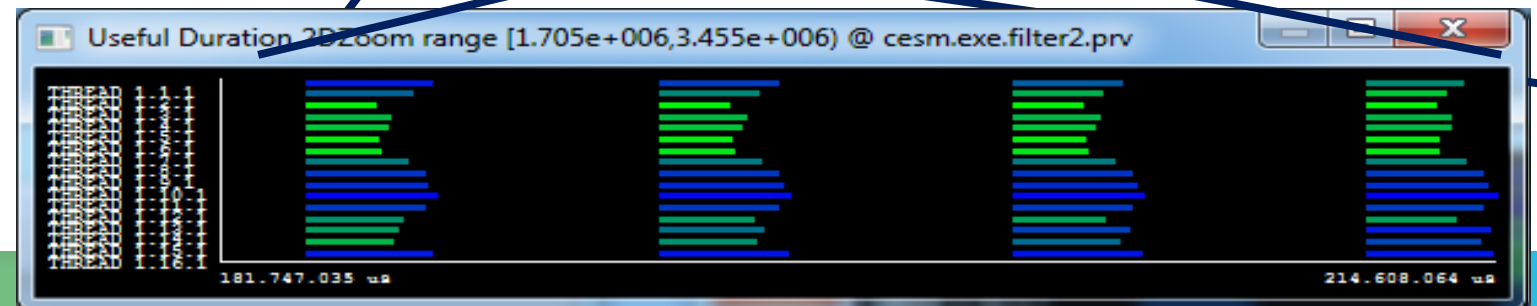
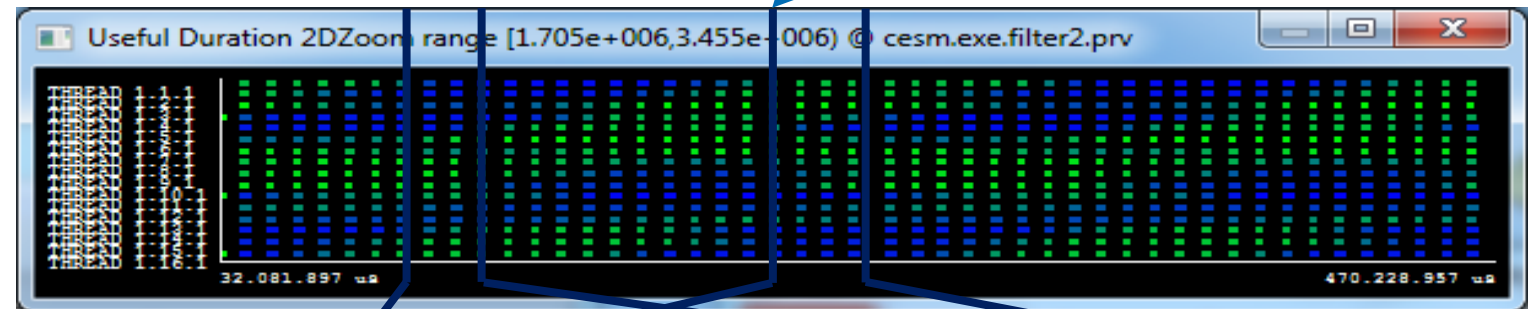
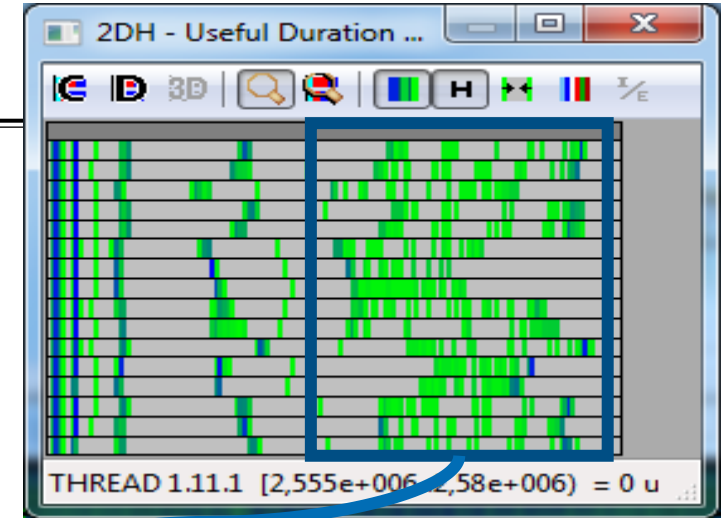
# Analyzing variability through histograms and timelines

- By the way: six months later ....



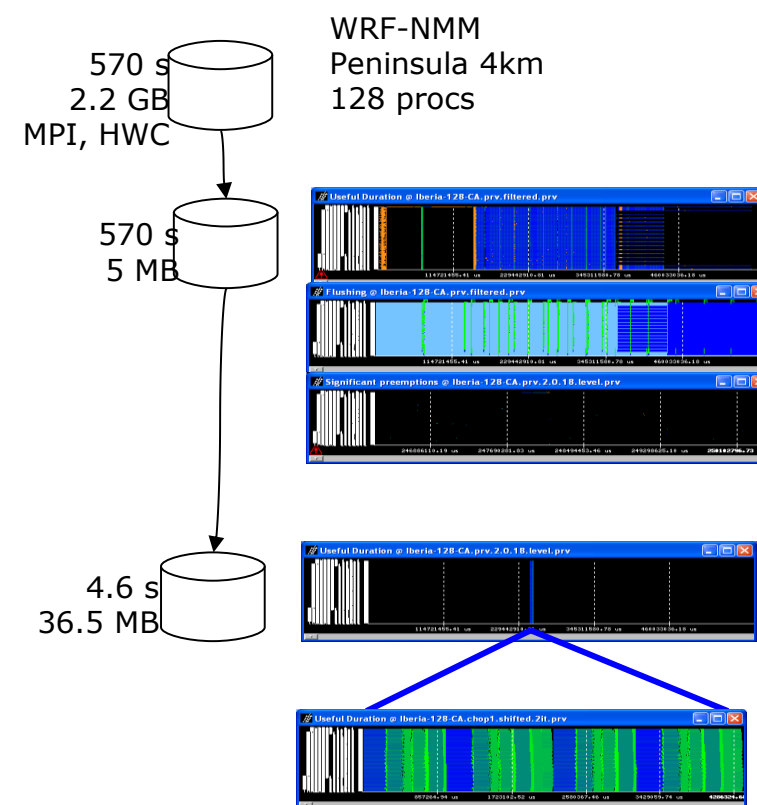
# Variability ... is everywhere

- 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

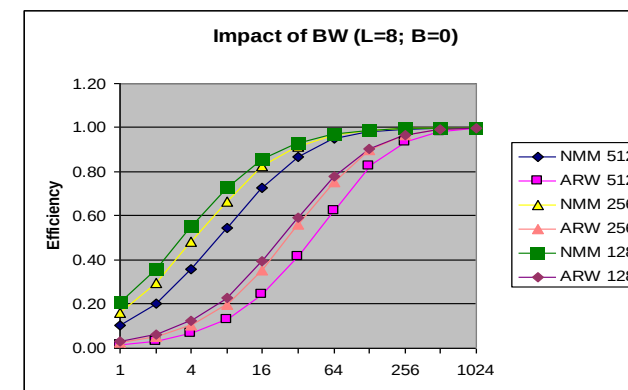
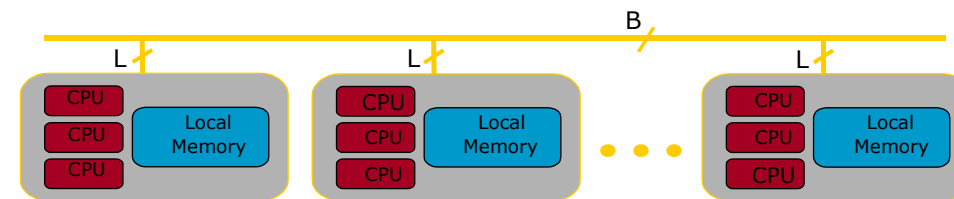
- Data handling/summarization capability
- Filtering
  - Subset of records in original trace
  - By duration, type, value,...
  - Filtered trace IS a paraver trace and can be analysed with the same cfgs (as long as needed data kept)
- Cutting
  - All records in a given time interval
  - Only some processes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



# Dimemas

## Dimemas: Coarse grain, Trace driven simulation

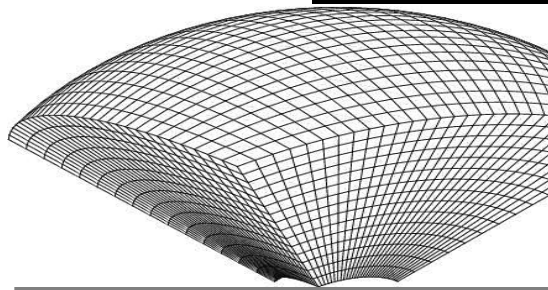
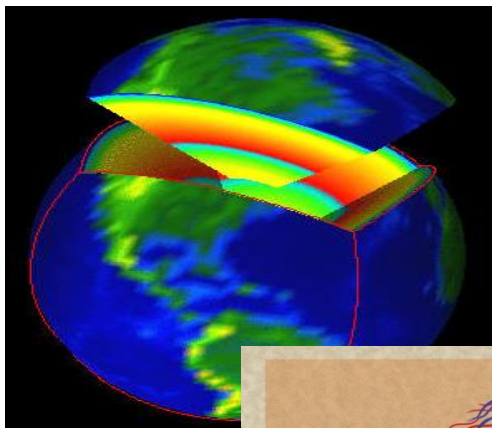
- Simulation: Highly non linear model
  - MPI protocols, resources contention...
- Parametric sweeps
  - On abstract architectures
  - On application computational regions
- What if analysis
  - Ideal machine (instantaneous network)
  - Estimating impact of ports to MPI+OpenMP/CUDA/...
  - Should I use asynchronous communications?
  - Are all parts of an app. 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)**

# Would I will benefit from asynchronous communications?

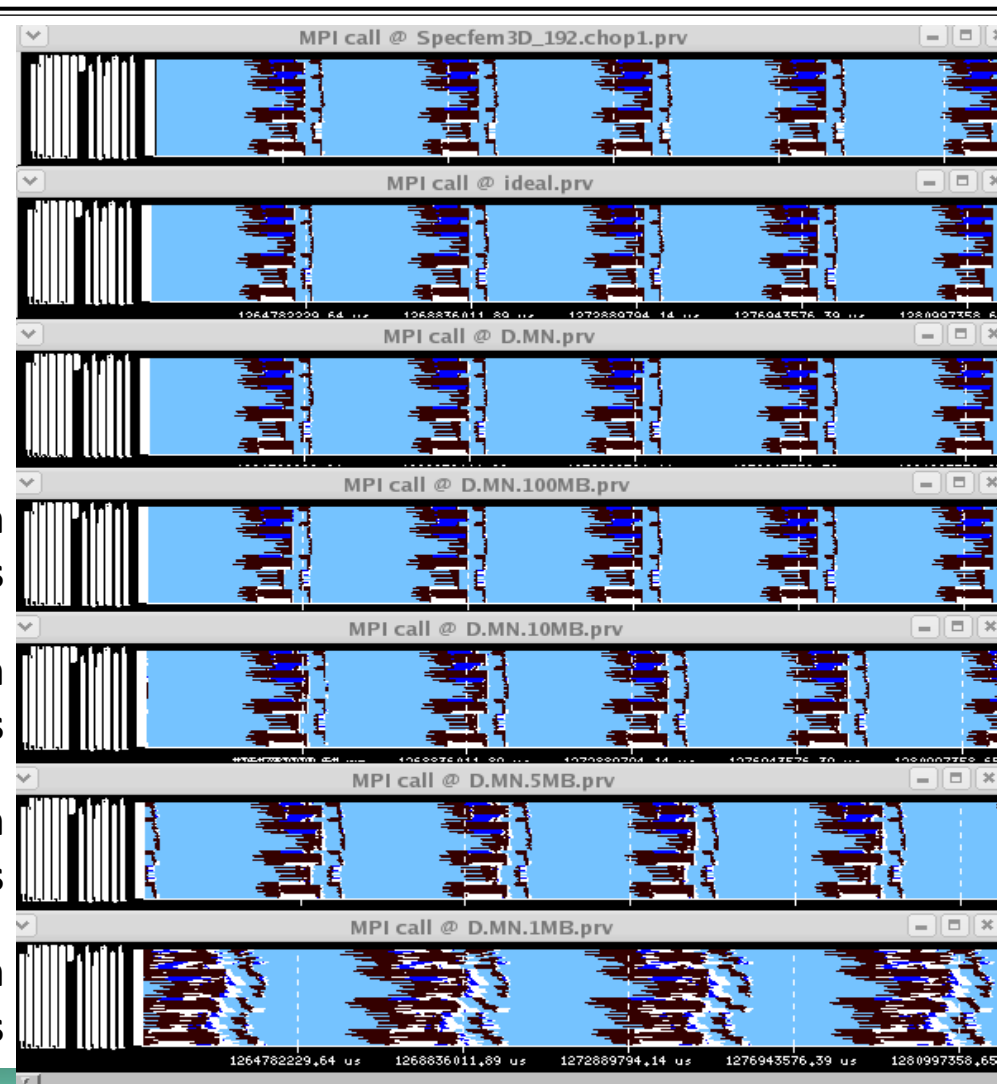
- SPECFEM3D



Courtesy Dimitri Komatitsch



Real  
Ideal  
Prediction  
MN  
Prediction  
100MB/s  
Prediction  
10MB/s  
Prediction  
5MB/s  
Prediction  
1MB/s



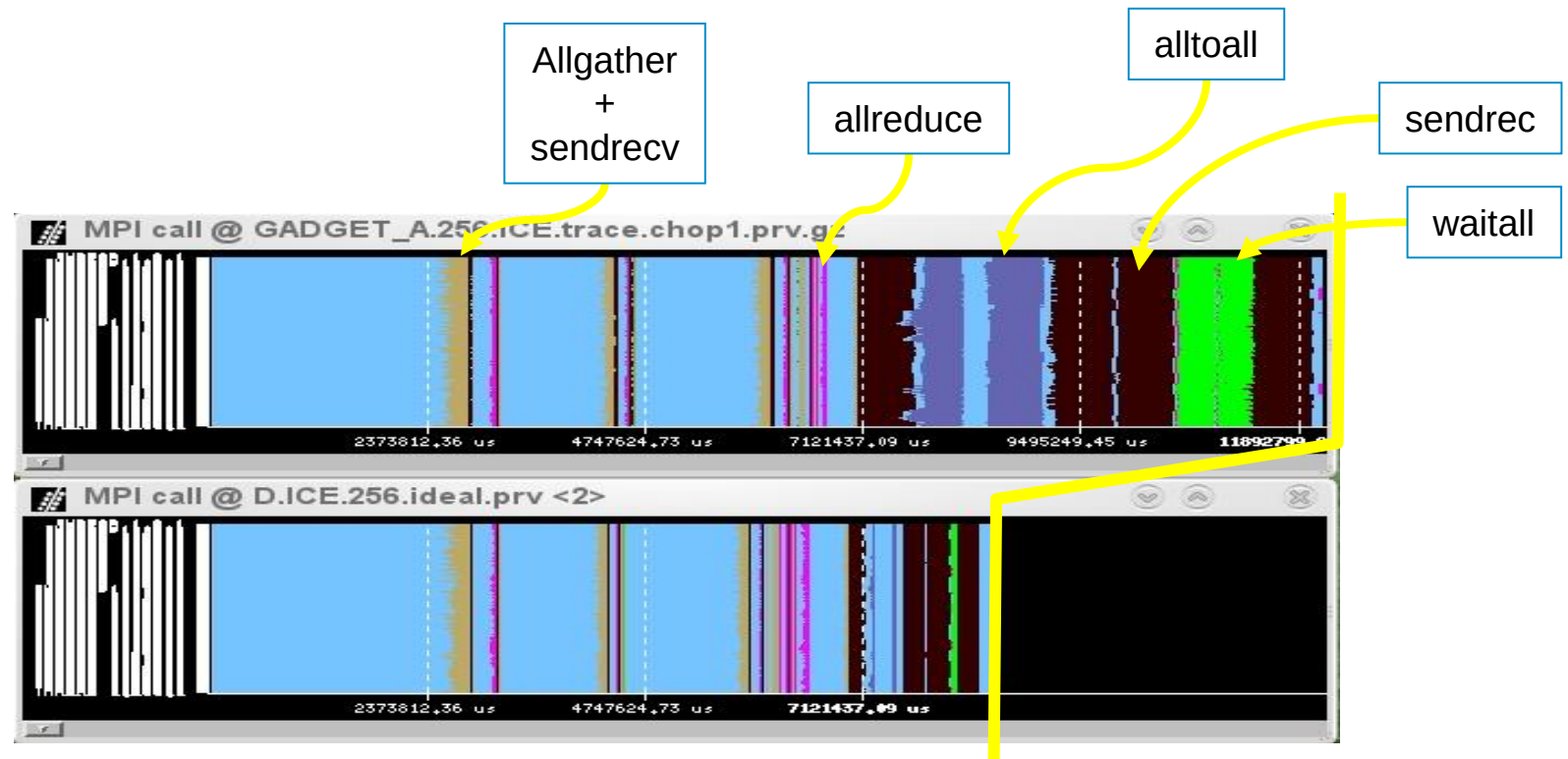
# Ideal machine

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

GADGET @ Nehalem cluster  
256 processes

Real  
run

Ideal  
network

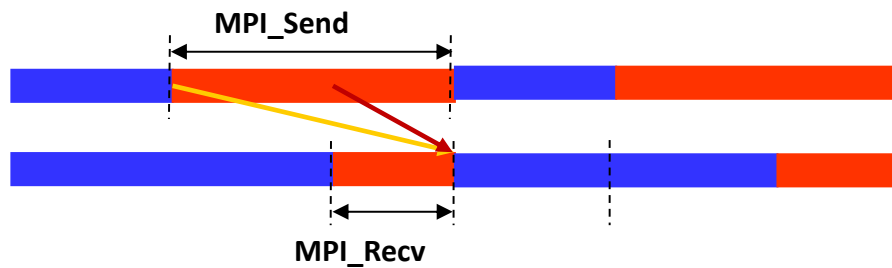


# Models

## Parallel efficiency model

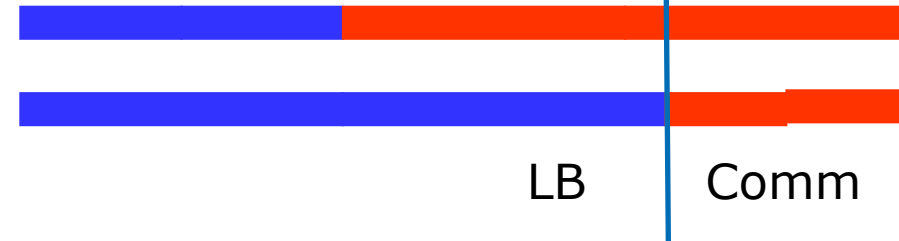
Computation

Communication



- Parallel efficiency = LB eff \* Comm eff

Do not blame MPI



2DP - MPI call profile @ trace\_24h\_atmos\_symbols.cho...

|                | Outside MPI | MPI_Recv | MPI_Isend  | MPI_Irecv |
|----------------|-------------|----------|------------|-----------|
| THREAD 1.130.1 | 87,95 %     | 9,09 %   | 0,00 %     | 0,02 %    |
| THREAD 1.131.1 | 88,16 %     | 9,09 %   | 0,00 %     | 0,02 %    |
| THREAD 1.132.1 | 88,18 %     | 9,09 %   | 0,00 %     | 0,02 %    |
| THREAD 1.133.1 | 88,18 %     | 9,09 %   | 0,00 %     | 0,02 %    |
| Total          | 9,309,74 %  | 306,53 % | 1,411,18 % | 3,83 %    |
| Average        | 69,00 %     | 2,30 %   | 10,69 %    | 0,03 %    |
| Maximum        | 88,18 %     | 67,62 %  | 54,97 %    |           |
| Minimum        | 30,67 %     | 0,00 %   | 0,00 %     |           |
| StDev          | 15,27 %     | 6,06 %   | 21,40 %    | 0,00 %    |
| Avg/Max        | 0,77        | 0,03     | 0,19       | 0,81      |

η

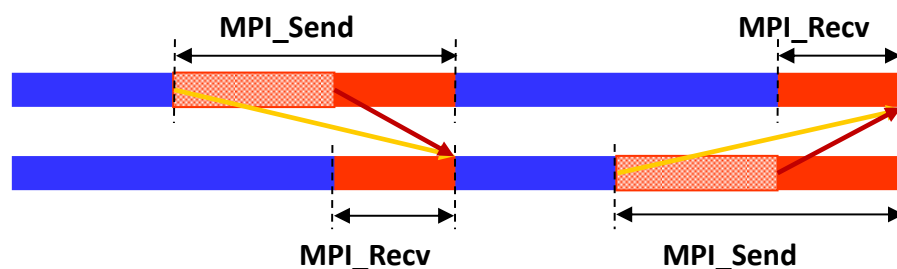
CommEff

LB

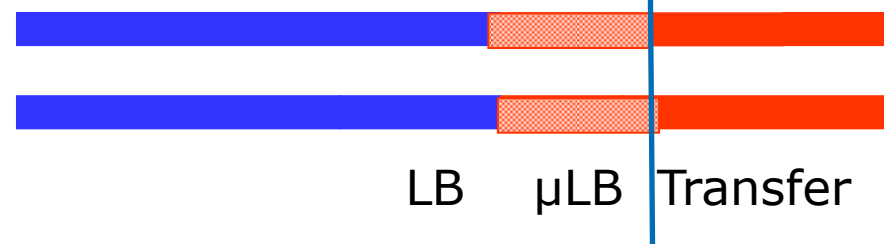
## Parallel efficiency refinement: $LB * \mu LB * \text{Transfer}$

Computation

Communication

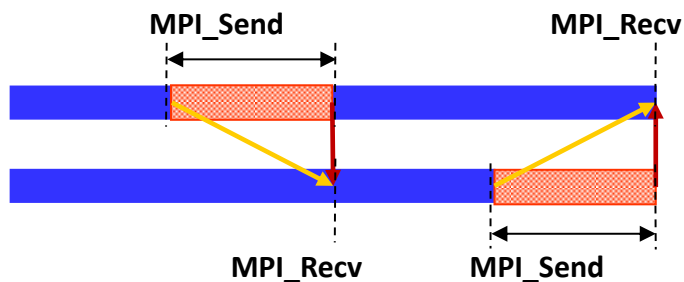


Do not blame MPI



$$LB=1$$

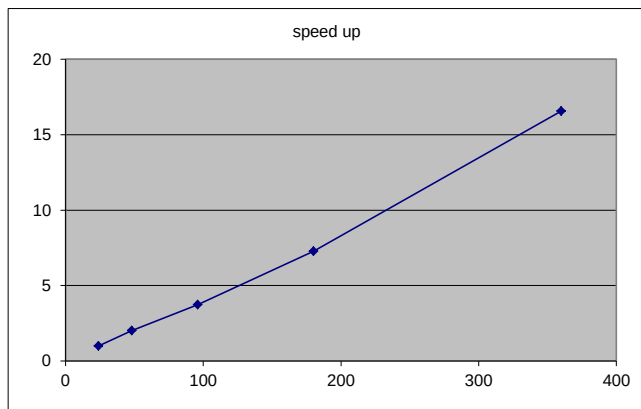
- Serializations / dependences ( $\mu LB$ )
- Dimemas ideal network  $\rightarrow$  Transfer (efficiency) = 1



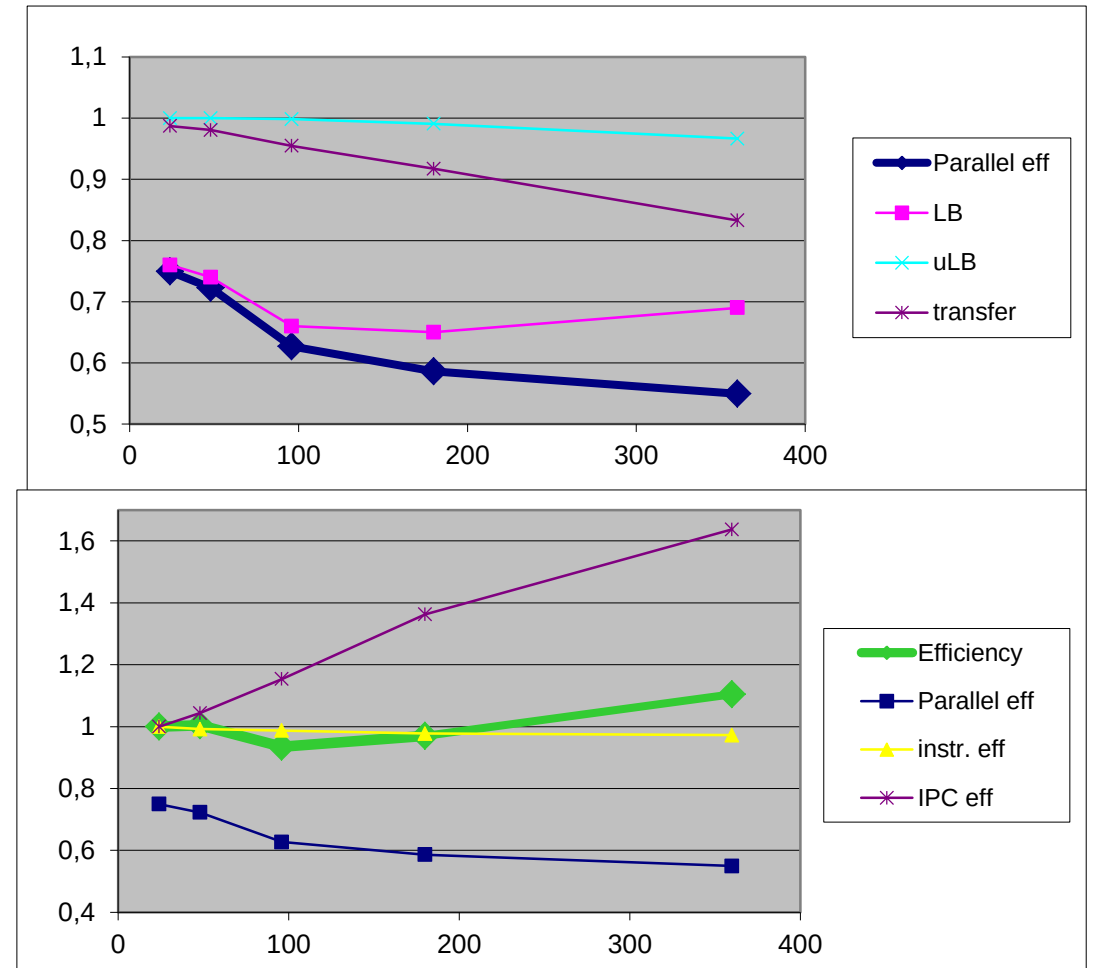
# Why scaling?

CG-POP mpi2s1D - 180x120

Good scalability !!  
Should we be happy?

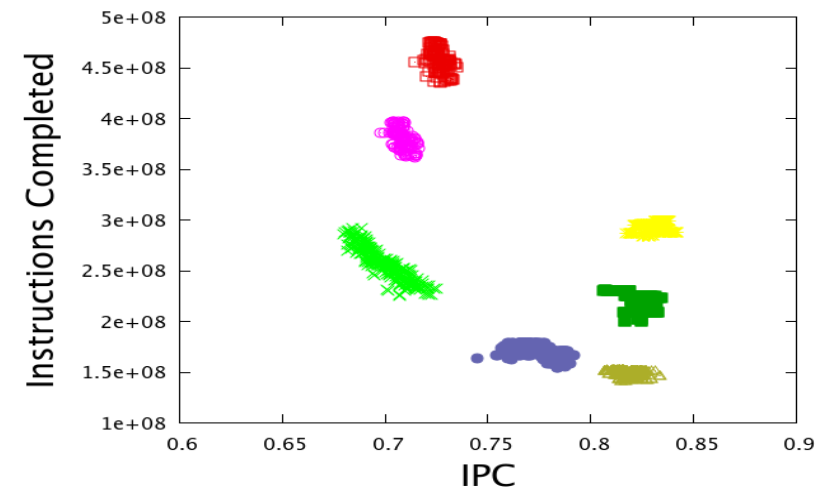
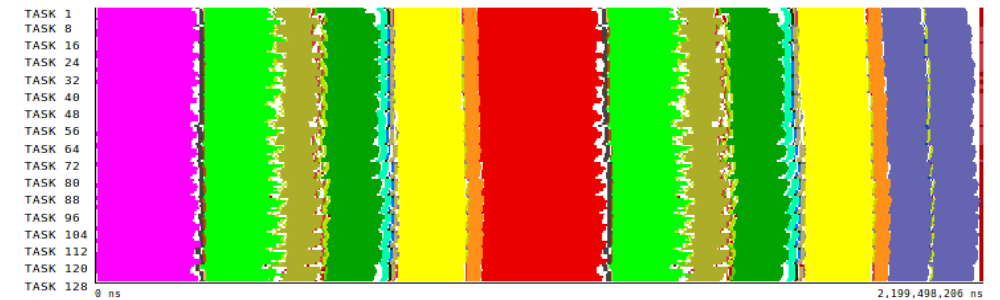
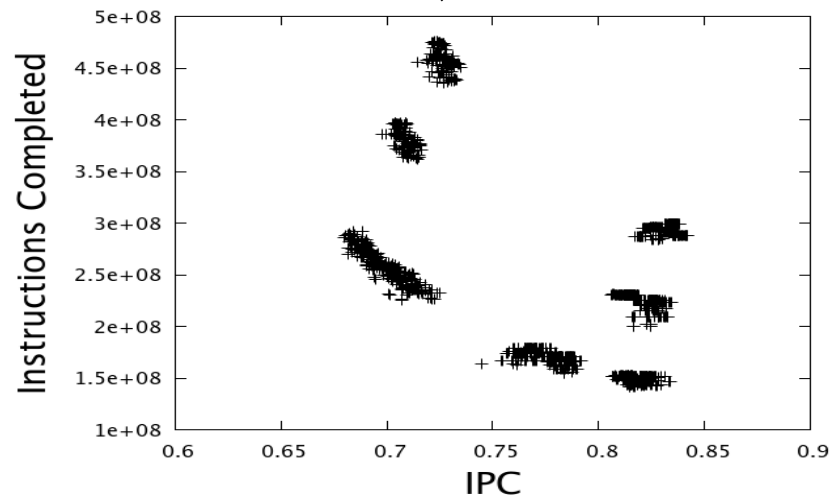
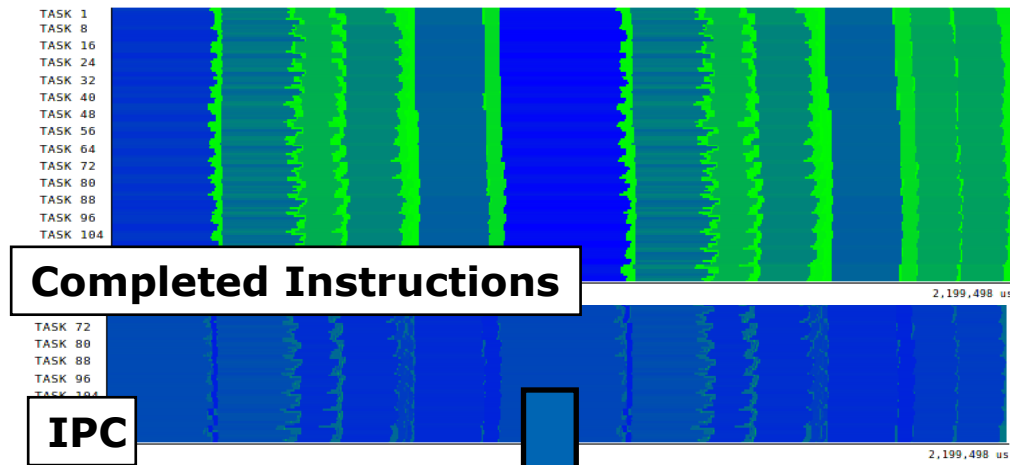


$$\eta_{\parallel} = LB * Ser * Trf$$



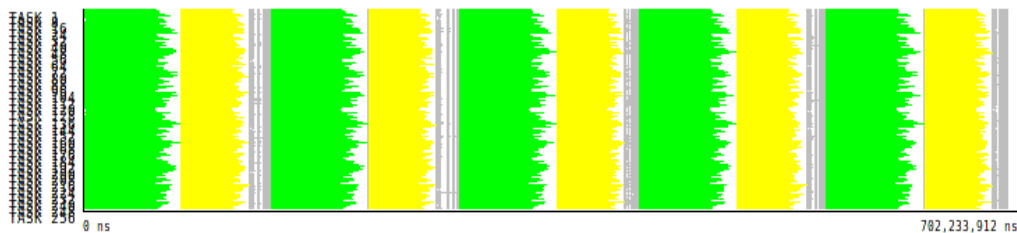
# Performance Analytics

# Using Clustering to identify structure



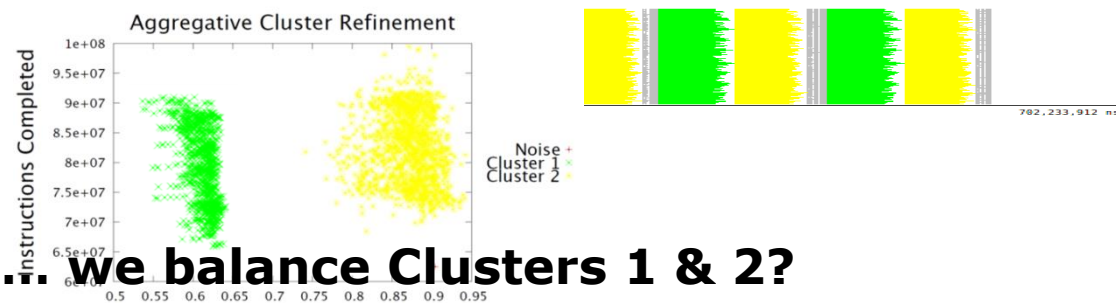
# Integrating models and analytics

What if ....



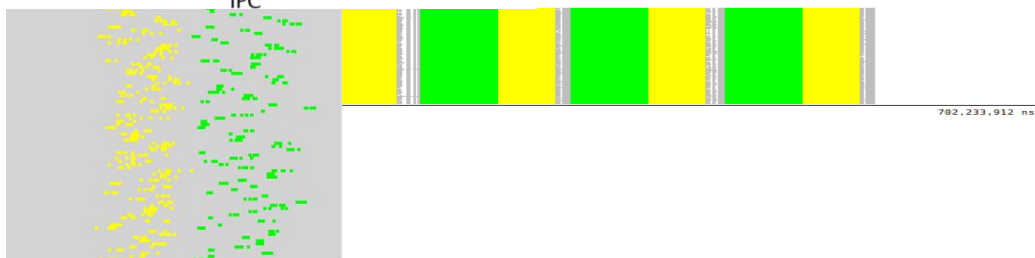
PEPC

... we increase the IPC of Cluster1?



13% gain

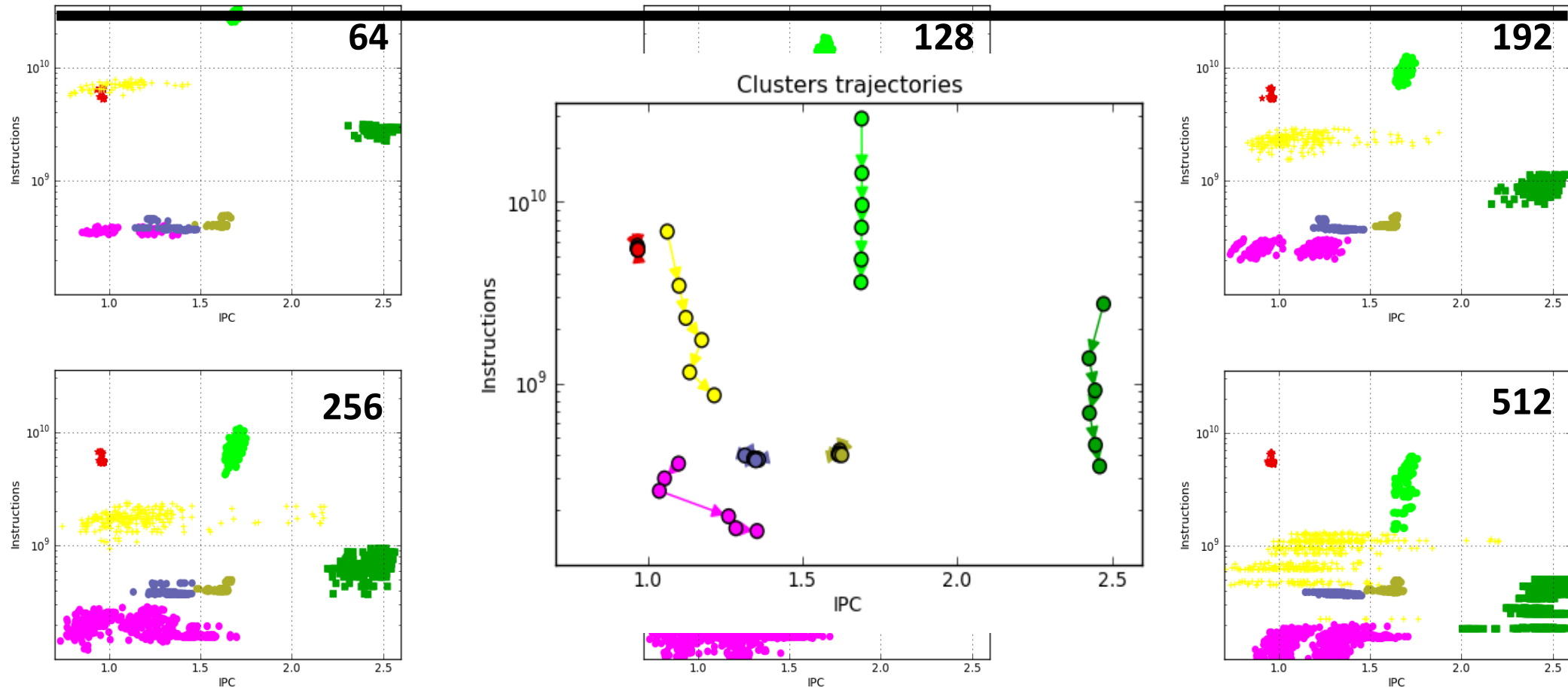
... we balance Clusters 1 & 2?



19% gain

# Tracking: scalability through clustering

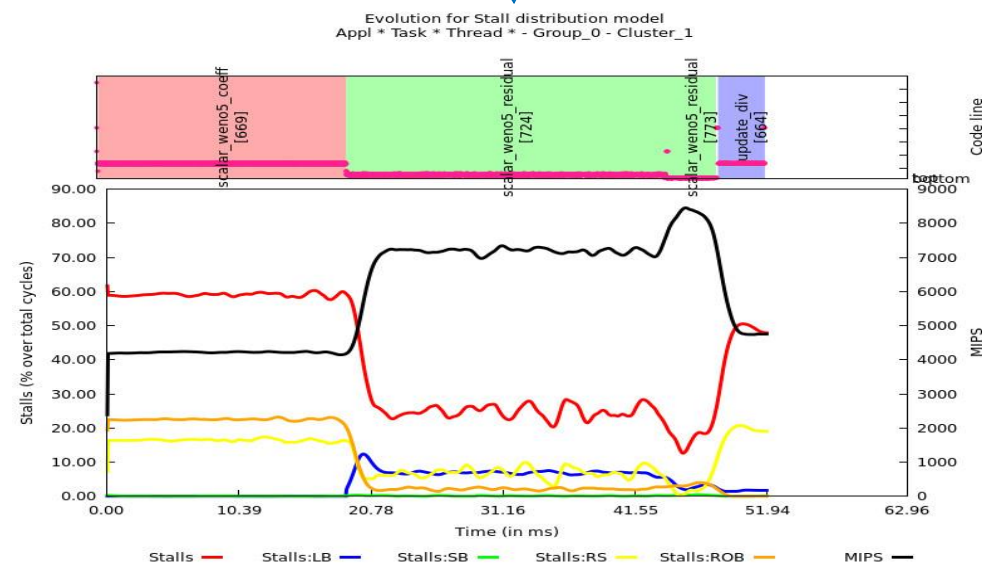
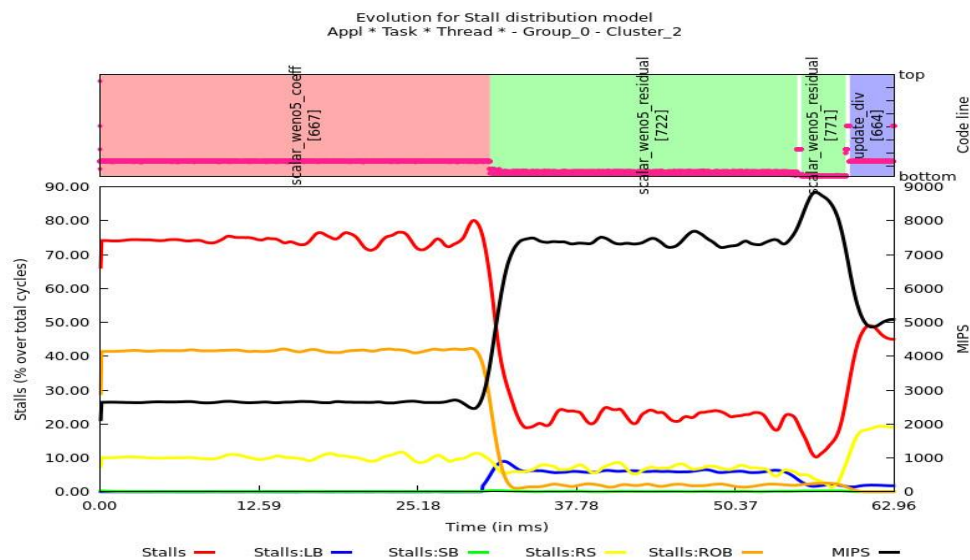
OpenMX (strong scale from 64 to 512 tasks)



# “Blind” optimization

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

Recommendation without access to source code



# Multicore effect

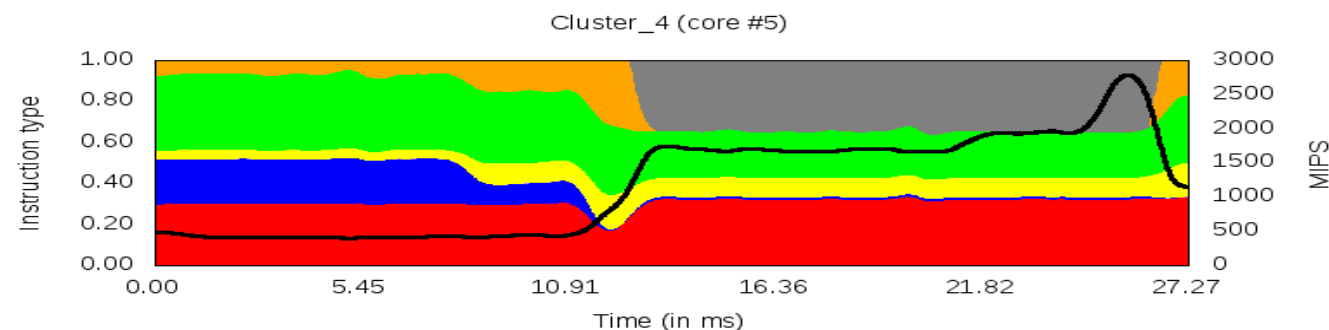
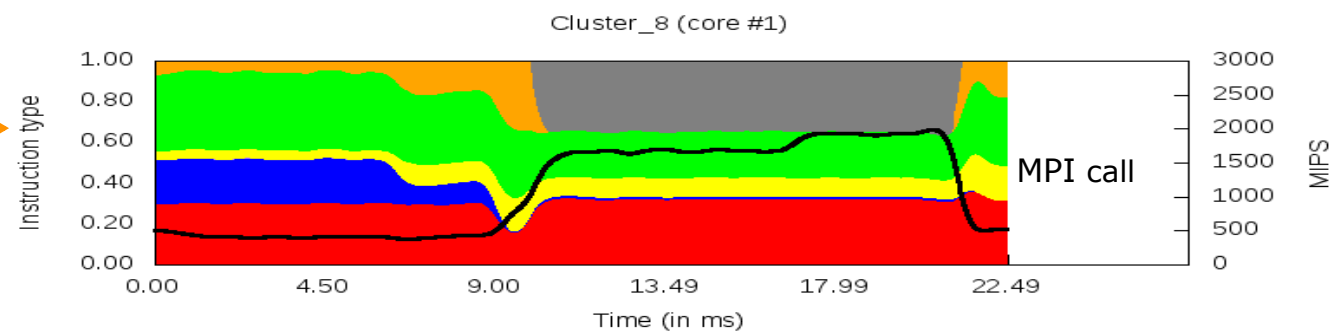
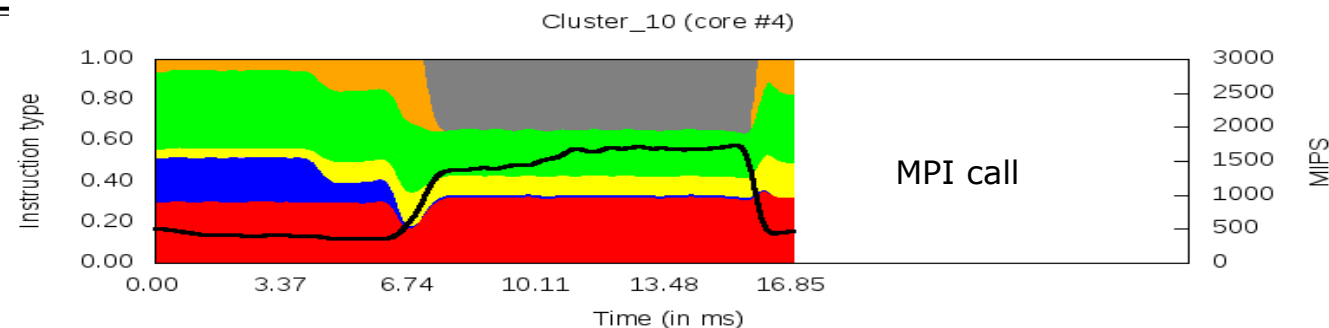
## ■ CG-POP: Unbalanced MPI application

- Same code
- Different duration
- Different performance

ClusterID @ cgpop.linux\_icc.180x120.chop2.clusterd.prv



Instruction mix model for the unbalanced CGPOP on different cores of the same hexacore chip



|    |      |           |        |           |        |        |            |
|----|------|-----------|--------|-----------|--------|--------|------------|
| LD | red  | uncond BR | purple | FP        | green  | Others | grey       |
| ST | blue | cond BR   | yellow | VEC sp+dp | orange | MIPS   | black line |

# 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?

Paraver Tutorial:  
Introduction to Paraver and Dimemas methodology

## BSC Tools web site

---

- [www.bsc.es/paraver](http://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

# Paraver 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                   |
| icon@mistral    | 79.86               | 84.02                    | 95.05                   |
| k-Wave@salomon  | 89.08               | 92.84                    | 95.96                   |
| fleur@claix     | 76.22               | 90.66                    | 84.07                   |

## Same code, different behaviour

| Code                   | Parallel efficiency | Communication efficiency | Load Balance efficiency |
|------------------------|---------------------|--------------------------|-------------------------|
| lulesh@mn3             | 90.55               | <b>99.22</b>             | 91.26                   |
| lulesh@leftraru        | <b>69.15</b>        | 99.12                    | <b>69.76</b>            |
| 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                   |
| lulesh@thunderX        | 73.96               | 97.56                    | 75.81                   |
| lulesh@jetson          | 75.48               | <b>88.84</b>             | 84.06                   |
| lulesh@claix           | 77.28               | 92.33                    | 83.70                   |
| lulesh@claix (binding) | <b>91.58</b>        | 98.70                    | <b>92.78</b>            |