Performance analysis with

BSC-Tools

Jesus Labarta, Judit Gimenez
BSC

Performance analysis tools objective

Fly with instruments

Who can I blame?

Understand our systems

Insight

Identify most productive efforts

Performance analysis tools objective

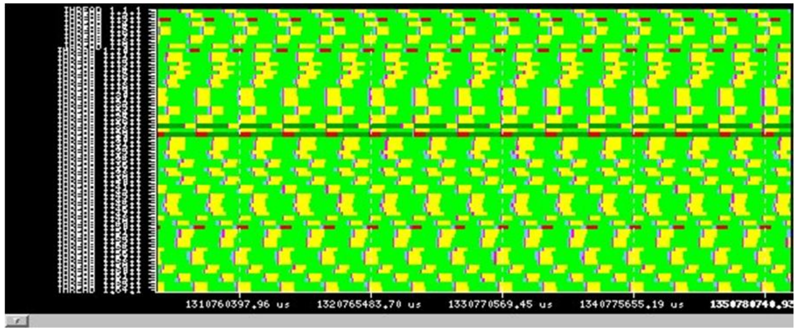
Help generate hypotheses

Help validate hypotheses

Qualitatively

Quantitatively

Our Tools

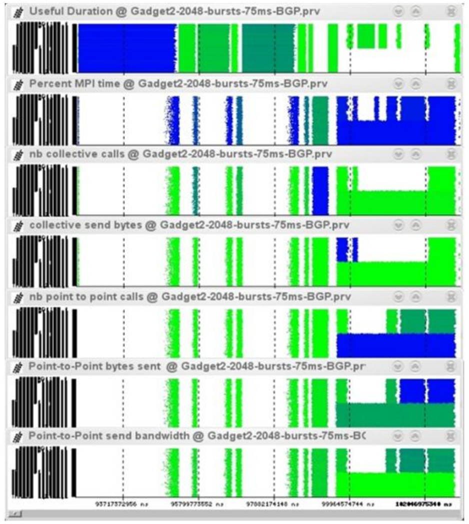


Since 1991

Based on traces

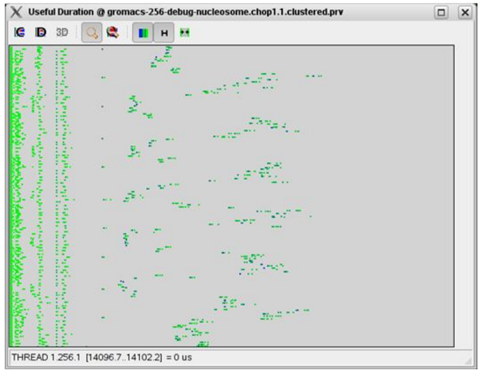
Open Source

http://www.bsc.es/paraver



Core tools:

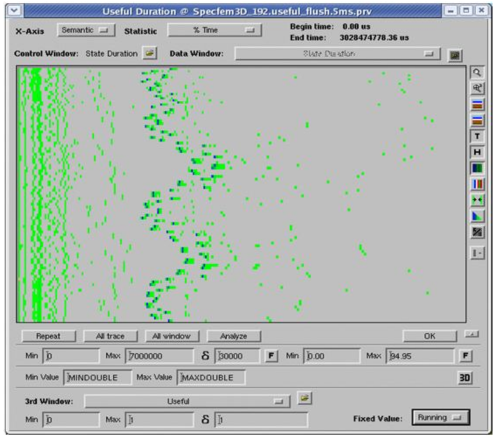
- Paraver – offline trace analysis
- Dimemas – message passing simulator
- Extrae – instrumentation



Focus

- Detail, flexibility, intelligence

Why traces?



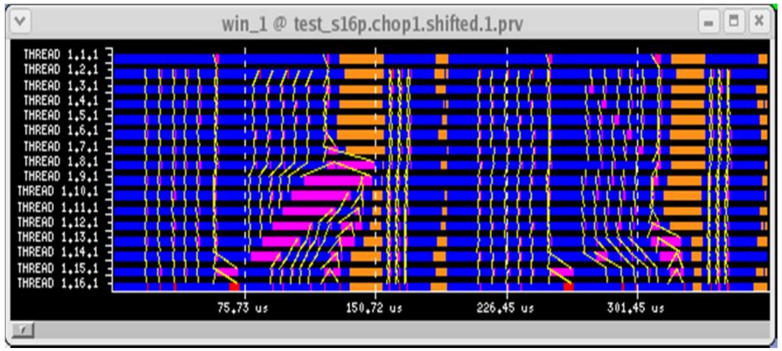
Detail and variability are important

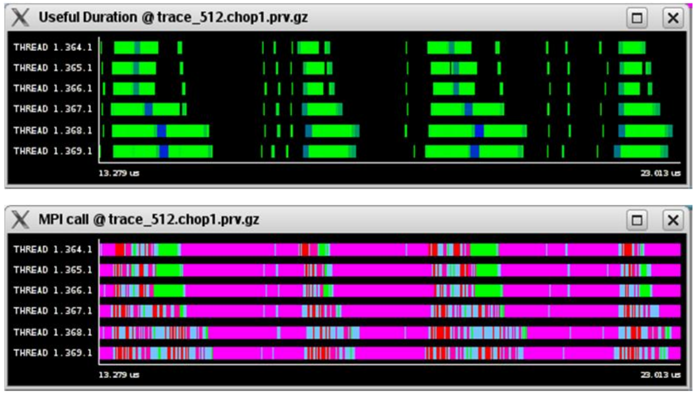
along time, across processors

Highly non-linear systems

microscopic effects may have macroscopic impact

Extremely useful to develop/test analysis techniques





Overview of capabilities of the environment

Analysis

- Trace analysis
- Scaling model
  - What if
  - Simple abstract model
- Structure detection
- HWC projection and CPI stack model
- Extreme detail at low overhead

Scalability

- Data selection and reduction
- Online

Outline

Extrae
Paraver
Dimemas

Scaling model
Structure detection
HWC analyses
- Projection and CPI Stack models
- Folding: Instrumentation + sampling

Scalability

Outline

Extrae
Paraver
Dimemas

Scaling model
Structure detection
HWC analyses
- Projection and CPI Stack models
- Folding

Scalability

Extrae – traced information

Parallel programing model runtime

MPI, OpenMP, Pthreads, StarSs, ...

Counters

CPU counters – PAPI (standard + native)

Network counters (GM, MX) - at flushes

OS counters

Links to source

Callstack at MPI calls

User functions selected – default none

Periodic samples

PAPI counters + callstack

User events

Outline

Extrae

Paraver

Dimemas

Scaling model

Structure detection

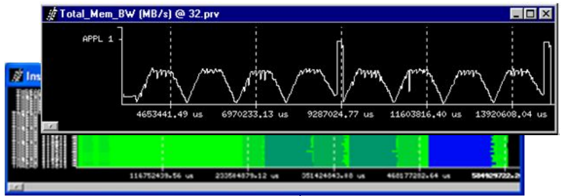
HWC analyses

- Projection and CPI Stack models
- Folding

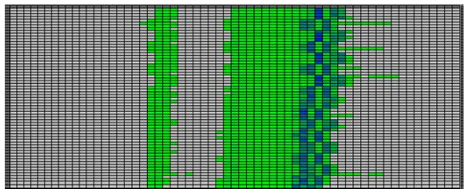
Scalability

Paraver – Performance data browser

Raw data



Timelines



2/3D tables
(Statistics)

Goal = Flexibility

No semantics

Programmable

Configuration files

Distribution

Your own

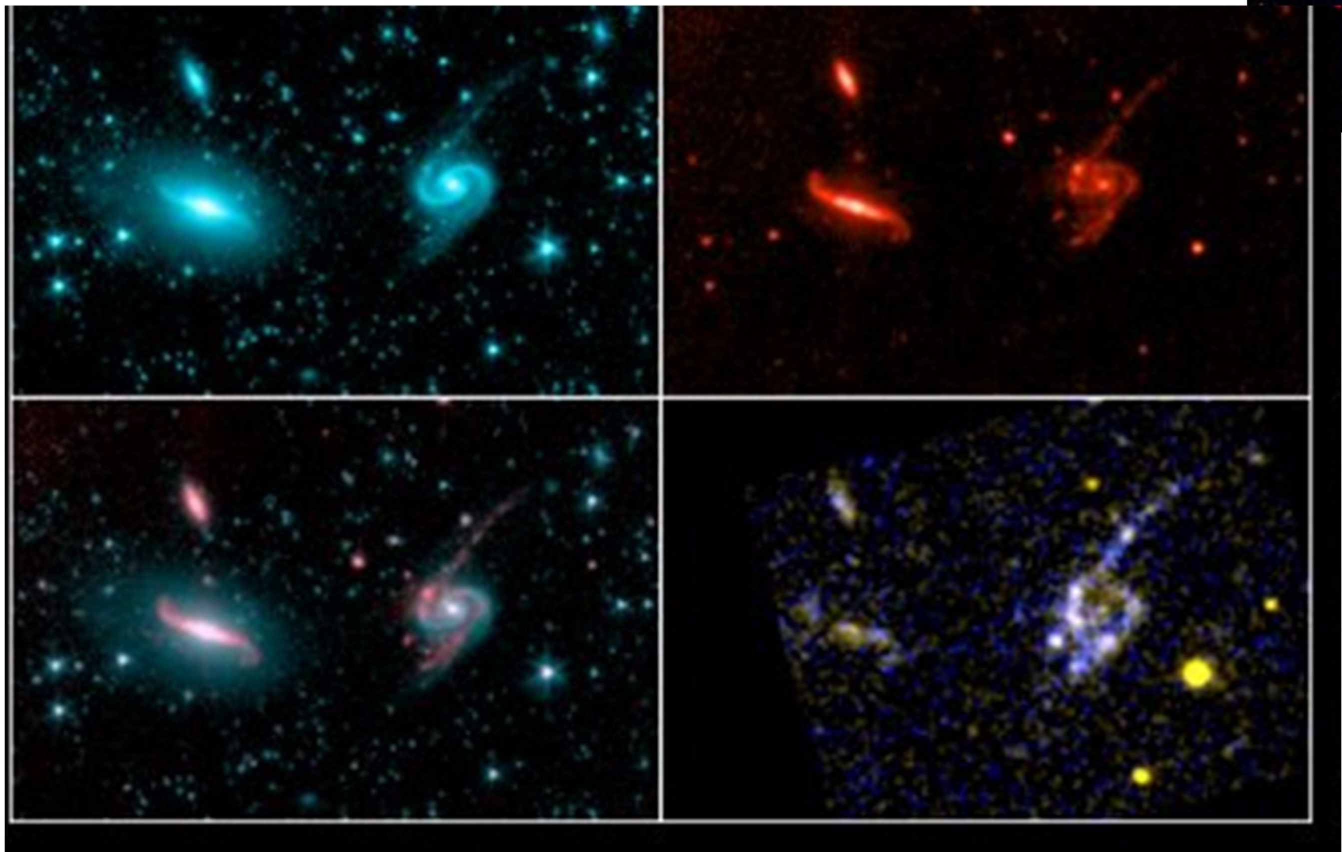
Comparative analyses

Multiple traces

Synchronize scales

Multispectral imaging

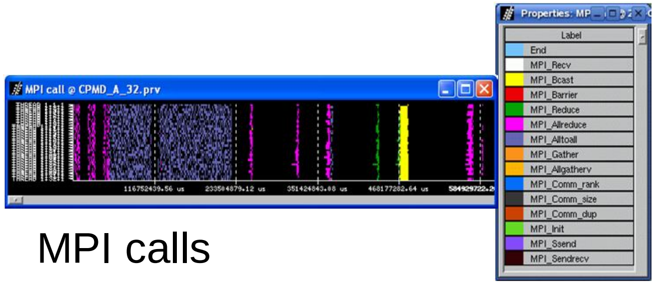
- Different looks at one reality
  - Different spectral bands (light sources and filters)
- Highlight different aspects



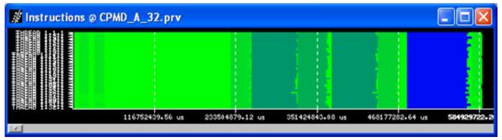
Views: Timelines

Raw events → Piece-wise constant functions of time → plots / colors

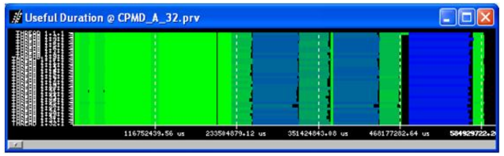
- Basic metrics



MPI calls

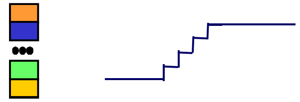


Instructions



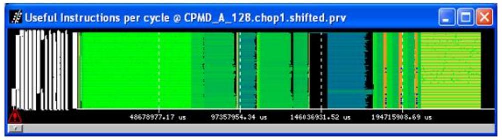
Useful duration

Color encoding

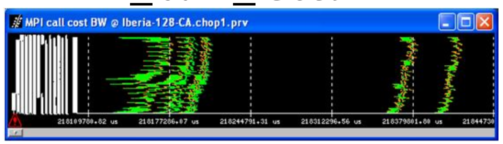


- Derived metrics

$$useful\_IPC = \frac{\#instr}{\#cycles} * useful$$



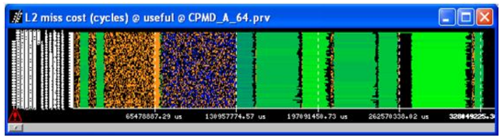
$$MPI\_call\_Cost = \frac{MPI\_call\_duration}{\#bytes}$$



- Models

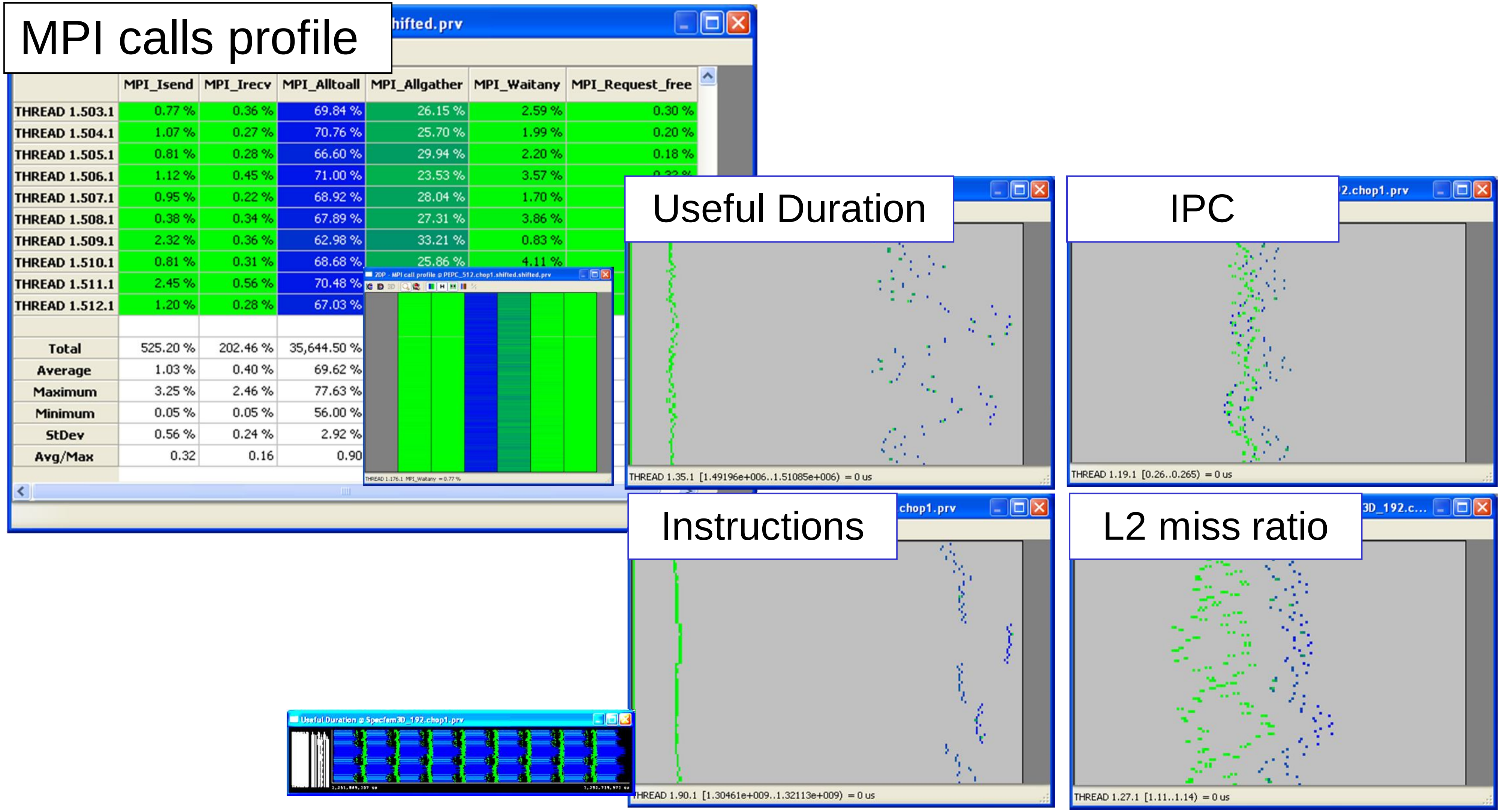
$$L2_{miss_{latency}} = \frac{cycles - instr / idealIPC}{L2misses}$$

$$preempted_{time} = elapsed - \frac{cycles}{clock_{freq}}$$



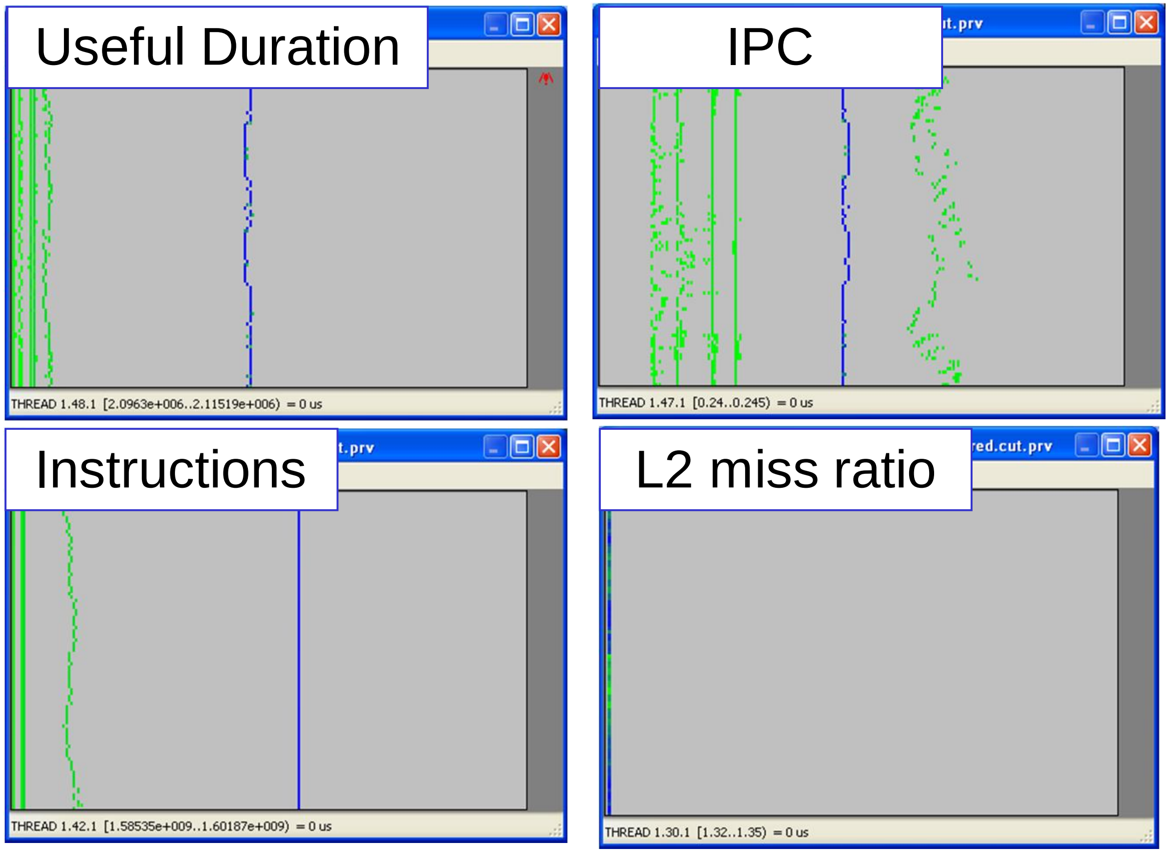
Tables: Profiles, histograms, correlations

Huge number of statistics computed from timelines



	MPI_Isend	MPI_Irecv	MPI_Alltoall	MPI_Allgather	MPI_Waitany	MPI_Request_free
THREAD 1.503.1	0.77 %	0.36 %	69.84 %	26.15 %	2.59 %	0.30 %
THREAD 1.504.1	1.07 %	0.27 %	70.76 %	25.70 %	1.99 %	0.20 %
THREAD 1.505.1	0.81 %	0.28 %	66.60 %	29.94 %	2.20 %	0.18 %
THREAD 1.506.1	1.12 %	0.45 %	71.00 %	23.53 %	3.57 %	
THREAD 1.507.1	0.95 %	0.22 %	68.92 %	28.04 %	1.70 %	
THREAD 1.508.1	0.38 %	0.34 %	67.89 %	27.31 %	3.86 %	
THREAD 1.509.1	2.32 %	0.36 %	62.98 %	33.21 %	0.83 %	
THREAD 1.510.1	0.81 %	0.31 %	68.68 %	25.86 %	4.11 %	
THREAD 1.511.1	2.45 %	0.56 %	70.48 %			
THREAD 1.512.1	1.20 %	0.28 %	67.03 %			
Total	525.20 %	202.46 %	35,644.50 %			
Average	1.03 %	0.40 %	69.62 %			
Maximum	3.25 %	2.46 %	77.63 %			
Minimum	0.05 %	0.05 %	56.00 %			
StDev	0.56 %	0.24 %	2.92 %			
Avg/Max	0.32	0.16	0.90			

By the way: six months later ….



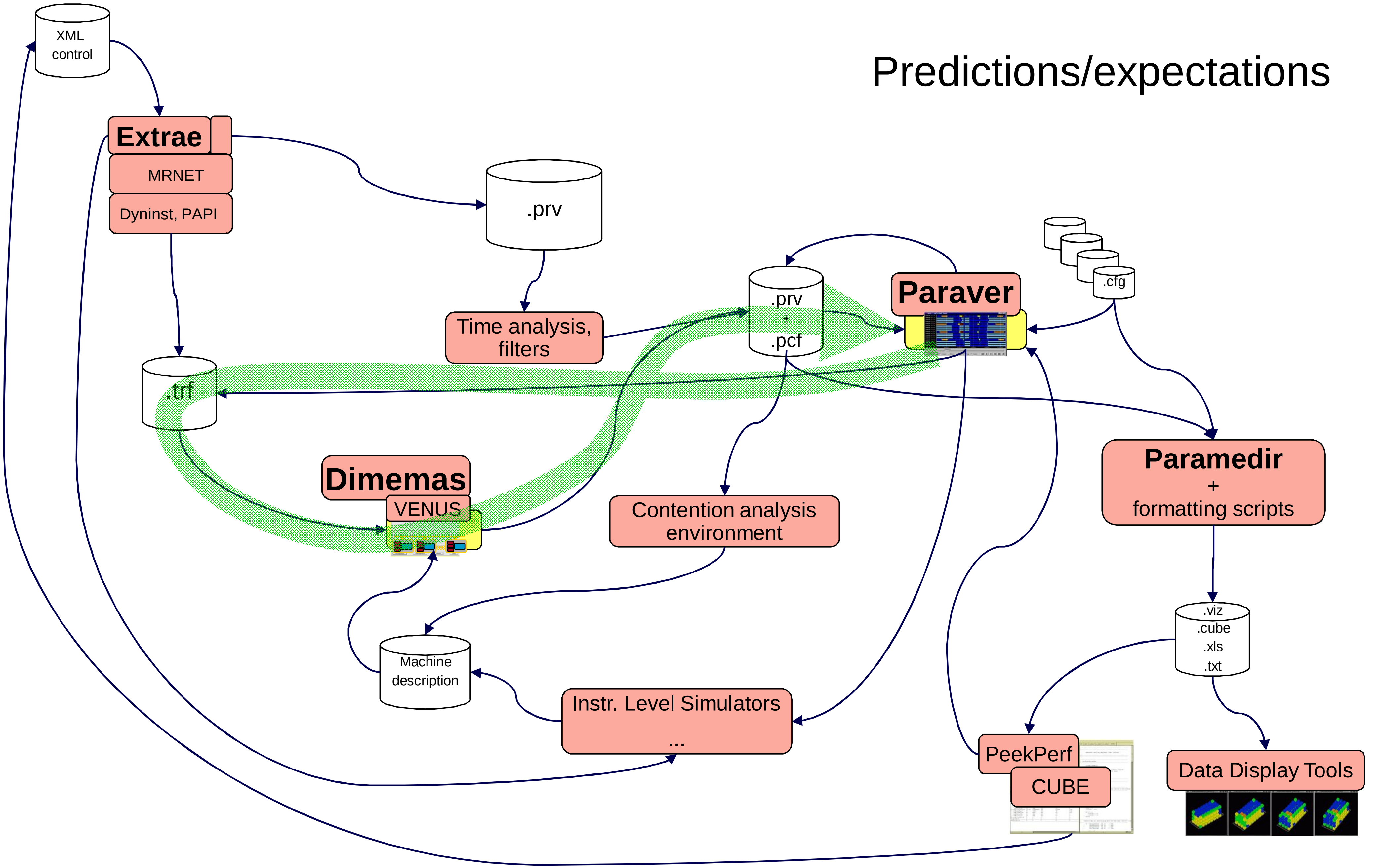
Outline

Extrae
Paraver
Dimemas

Scaling model
Structure detection
HWC analyses
  Projection and CPI Stack models
  Folding

Scalability

BSC-Tools Environment



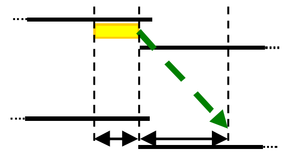
Dimemas: Coarse grain, Trace driven simulation

Simulation: Highly non linear model

Linear components

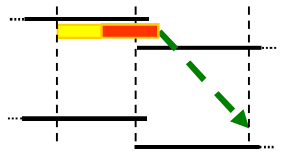
- Point to point communication

$$T = \frac{MessageSize}{BW} + L$$

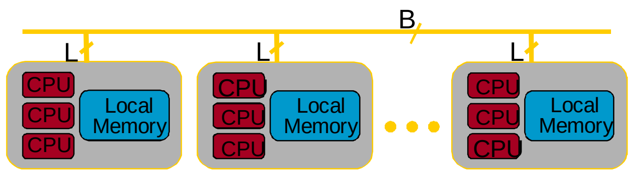


- Sequential processor performance
  - Global CPU speed
  - Per block/subroutine

Non linear components



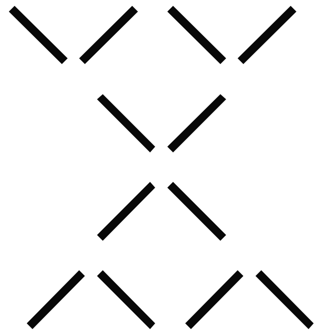
- Synchronization semantics
  - Blocking receives
  - Rendezvous
- Resource contention
  - CPU
  - Communication subsystem
    - links (half/full duplex), busses



Collective communication model

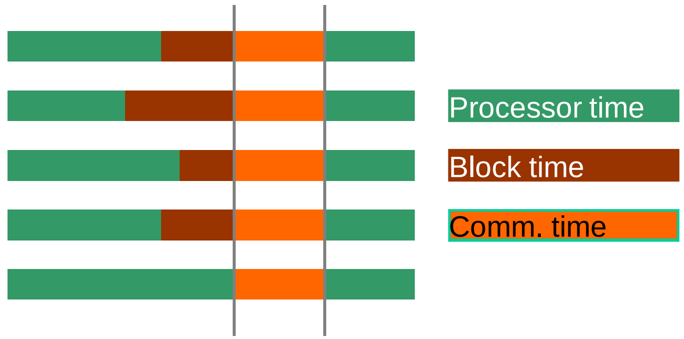
Generic model

- Barrier / Fan-in / Fan-out
- Cost of communication phase
  - Generic



  - Per call
    - Model factor
      - Lin / log / const
    - Size of message
      - Min over all processes
      - Avg over all processes
      - Max over all processes

Collective



Outline

Extrae

Paraver

Dimemas

Scaling model

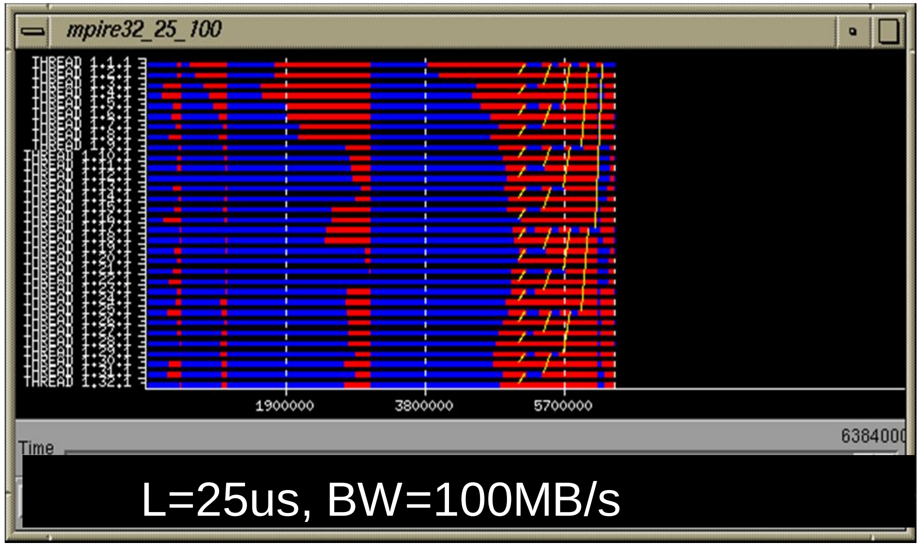
Structure detection

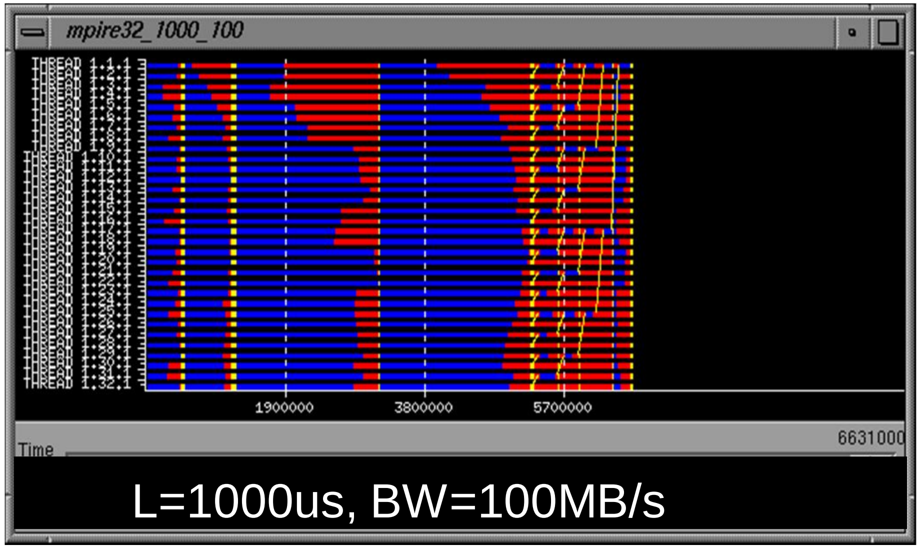
HWC analyses

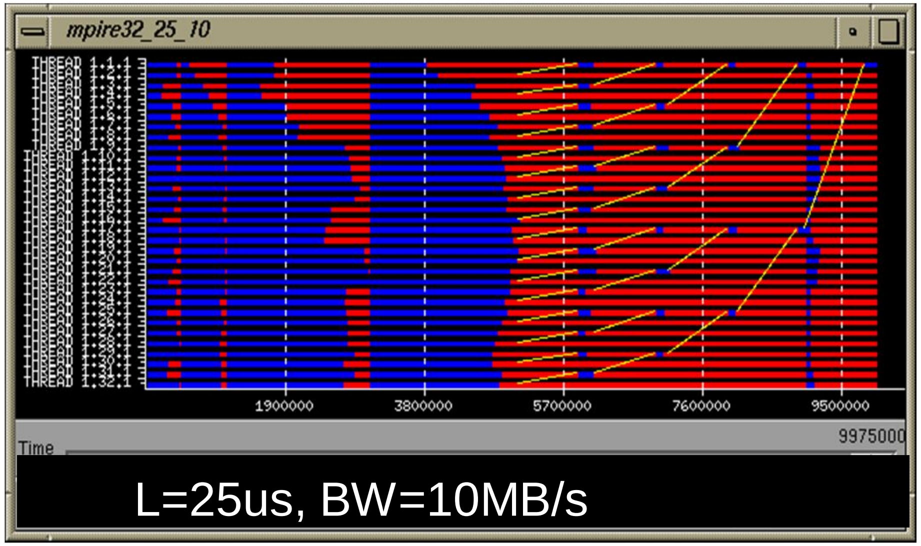
- Projection and CPI Stack models
- Folding

Scalability

MPIRE 32 tasks, no network contention



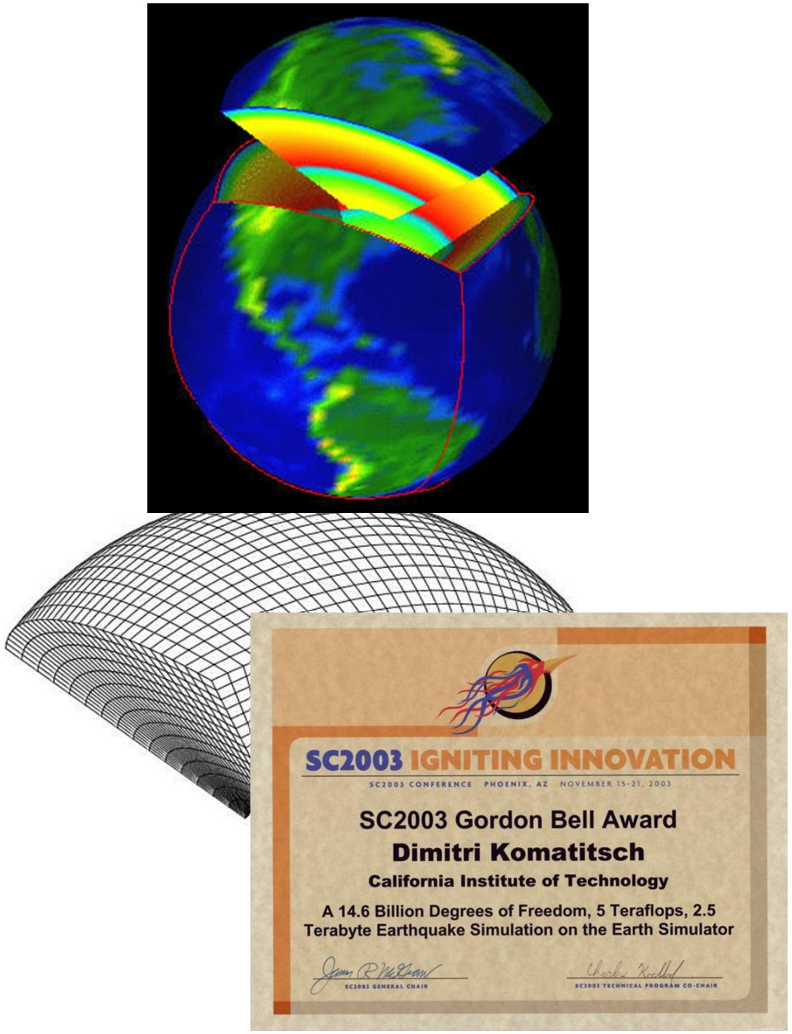




All windows same scale

Insight → advice appropriate direction

SPECFEM3D – asynchronous communication?



Real

ideal

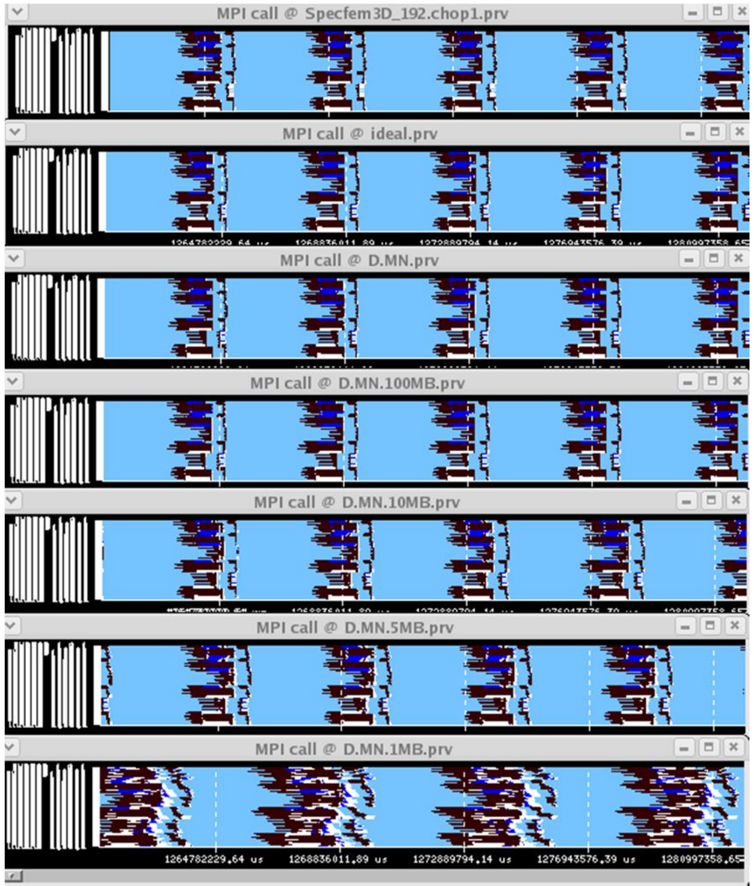
MN prediction

Prediction 100MB/s

Prediction 10MB/s

Prediction 5MB/s

Prediction 1MB/s



Courtesy Dimitri Komatitsch

Intrinsic application behavior

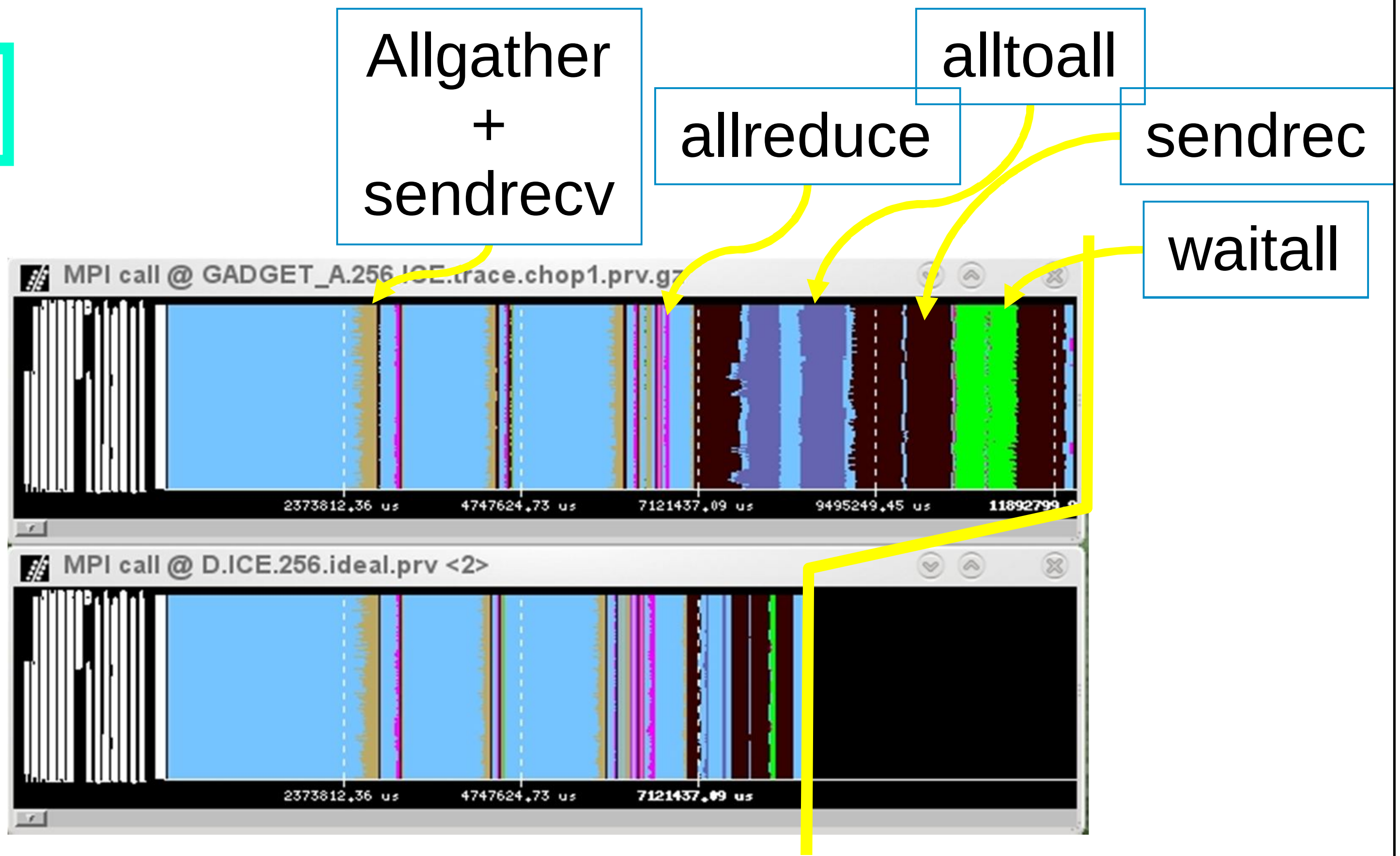
Load balanced and dependence problems?

BW = ∞,   L = 0

GADGET @ Nehalem cluster
256 processes

Real run

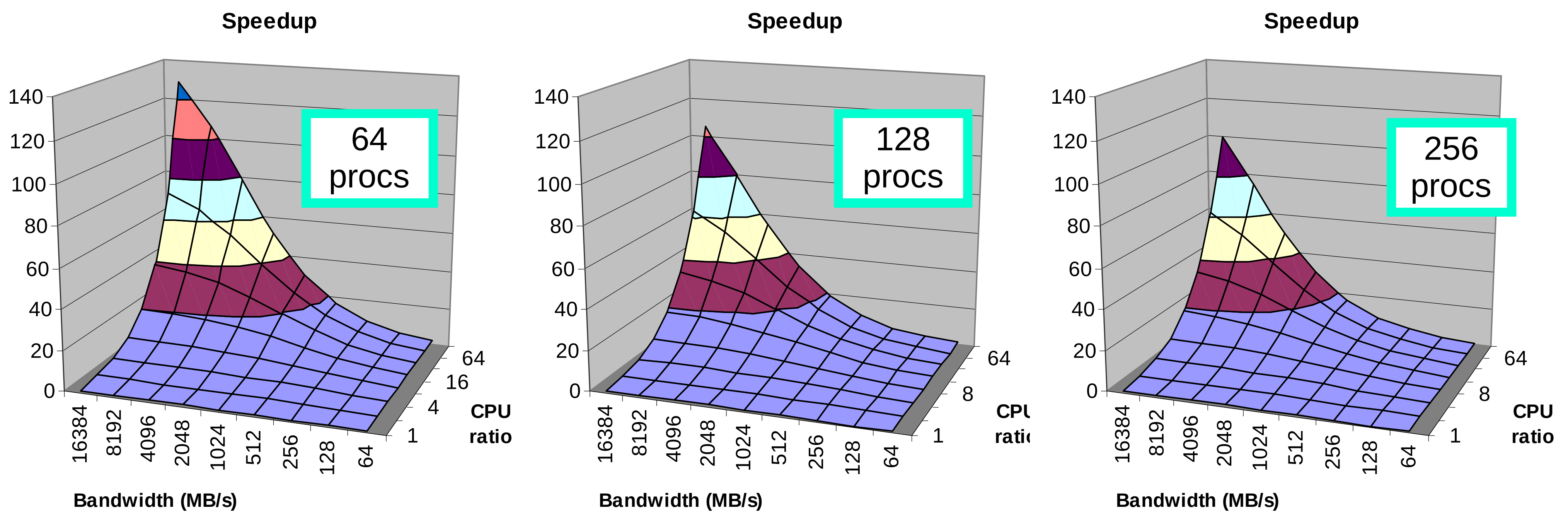
Ideal network



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) !!!!!

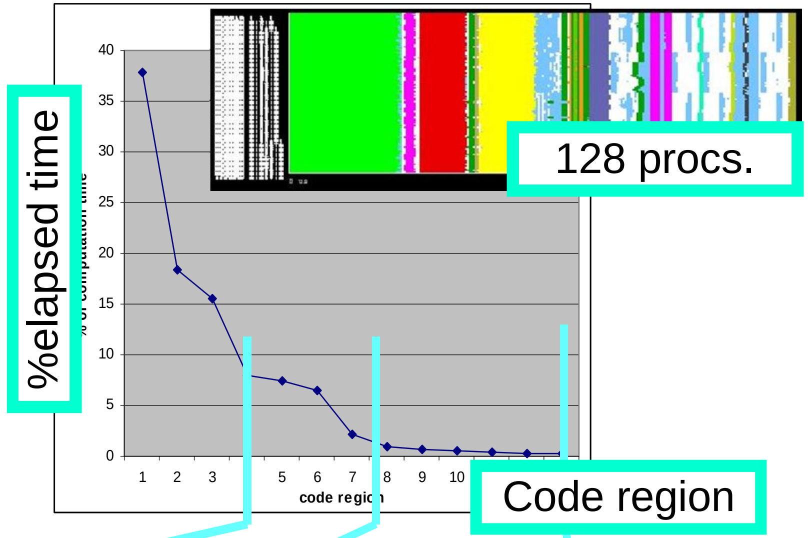


GADGET

What if: Potential of hybrid parallelization

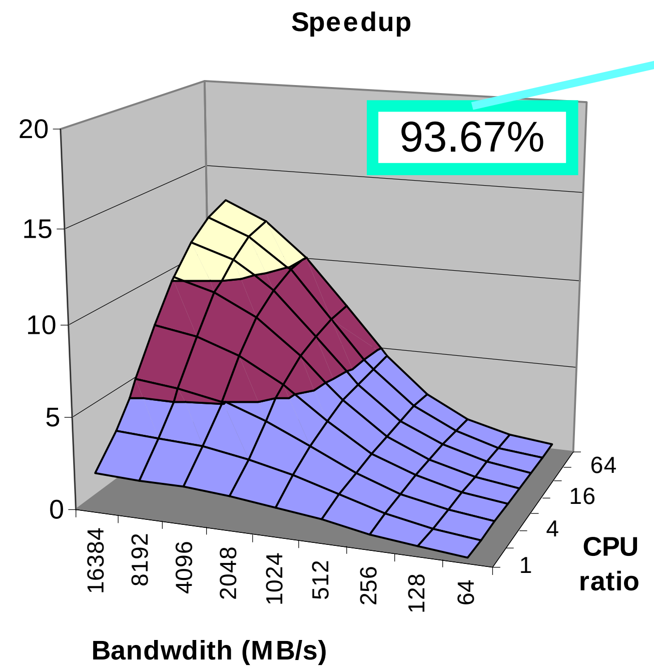
Hybrid/accelerator parallelization

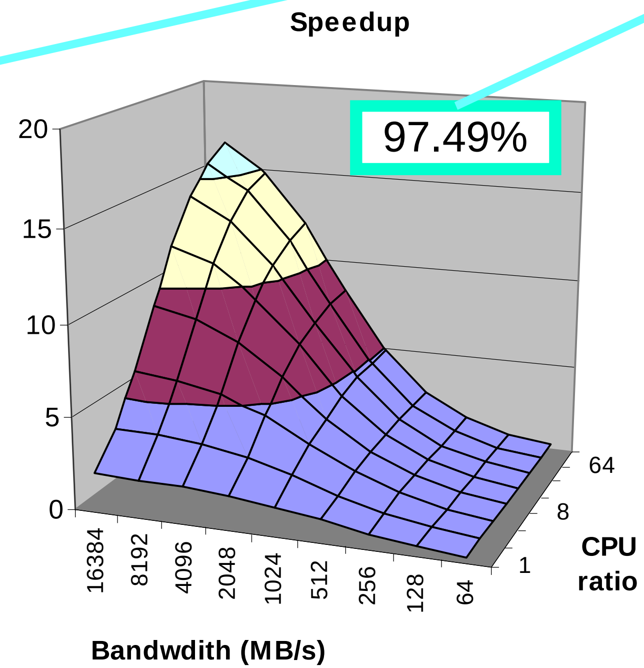
Speedup SELECTED regions by the CPUratio factor

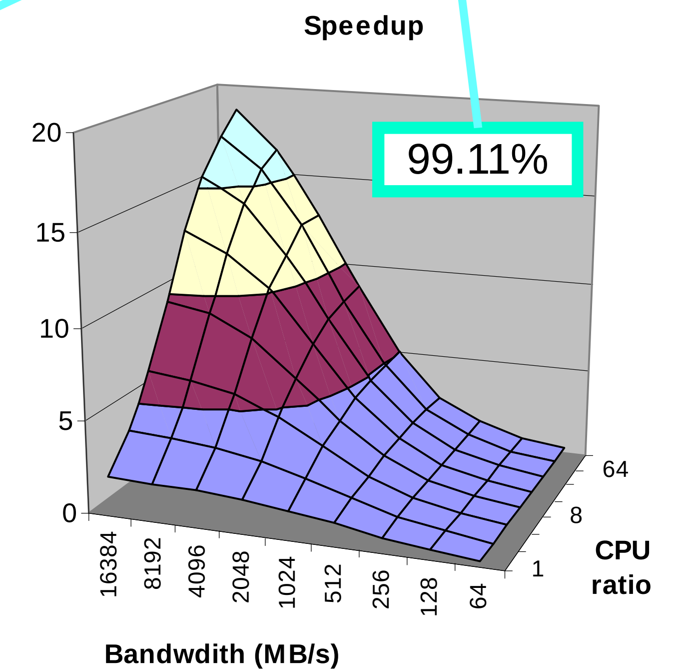


We do need to overcome the **hybrid Amdahl's law**

→ asynchrony + Load balancing mechanisms !!!







Presenting application performance

Factors modeling parallel efficiency

Load balance (LB)

Micro load balance (μLB) or **serialization**

Transfer

$$\eta = LB * \mu LB * Transfer$$

Factors describing serial behavior

Performance: **IPC**

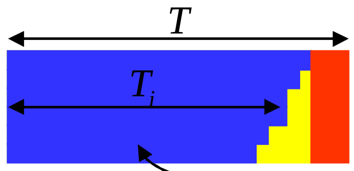
Scaling model

$$Sup = \frac{P}{P_0} * \frac{\eta}{\eta_0} * \frac{IPC}{IPC_0} * \frac{\#instr_0}{\#instr}$$

Scaling model

$$Sup = \frac{P}{P_0} * \frac{LB}{LB_0} * \frac{CommEff}{CommEff_0} * \frac{IPC}{IPC_0} * \frac{\#instr_0}{\#instr}$$

Directly from real execution metrics



$$eff_i = \frac{T_i}{T}$$

$$CommEff = \max(eff_i)$$

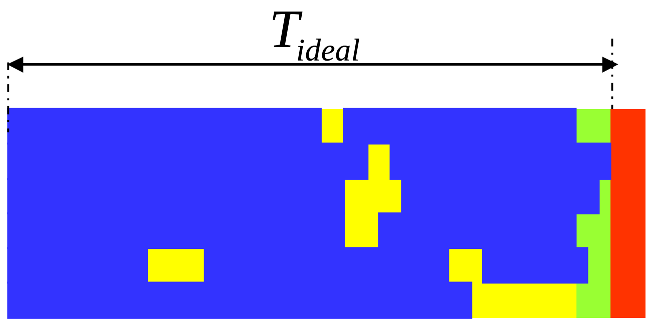
$$LB = \frac{\sum_{i=1}^{P} eff_i}{P * \max(eff_i)}$$

IPC
#instr

$$microLB = \frac{\max(T_i)}{T_{ideal}} \qquad Transfer = \frac{T_{ideal}}{T}$$

$$Sup = \frac{P}{P_0} * \frac{macroLB}{macroLB_0} * \frac{microLB}{microLB_0} * \frac{Transfer}{Transfer_0} * \frac{IPC}{IPC_0} * \frac{\#instr_0}{\#instr}$$

Requires Dimemas simulation



Migrating/local load imbalance
Serialization

Modeling Parallel performance

Almost Acceptable scalability

GADGET @ PRACE data set 1

Platform	Processors	Input	Iteration time (s)	Speedup	Relative efficiency
MN	64	A	78,71	1,00	1,00
MN	128	A	44,22	1,78	0,89
MN	256	A	22,78	3,46	0,86
BGP	64	A	250,41	1,00	1,00
BGP	128	A	131,78	1,90	0,95
BGP	256	A	79,10	3,17	0,79
ICE	64	A	40,41	1,00	1,00
ICE	128	A	21,65	1,87	0,93
ICE	256	A	12,15	3,32	0,83

Modeling Parallel performance

Good in BGP

Transfer Issue

Almost Acceptable scalability

Performs not so well

Fair load balance

GADGET @ PRACE data set 1

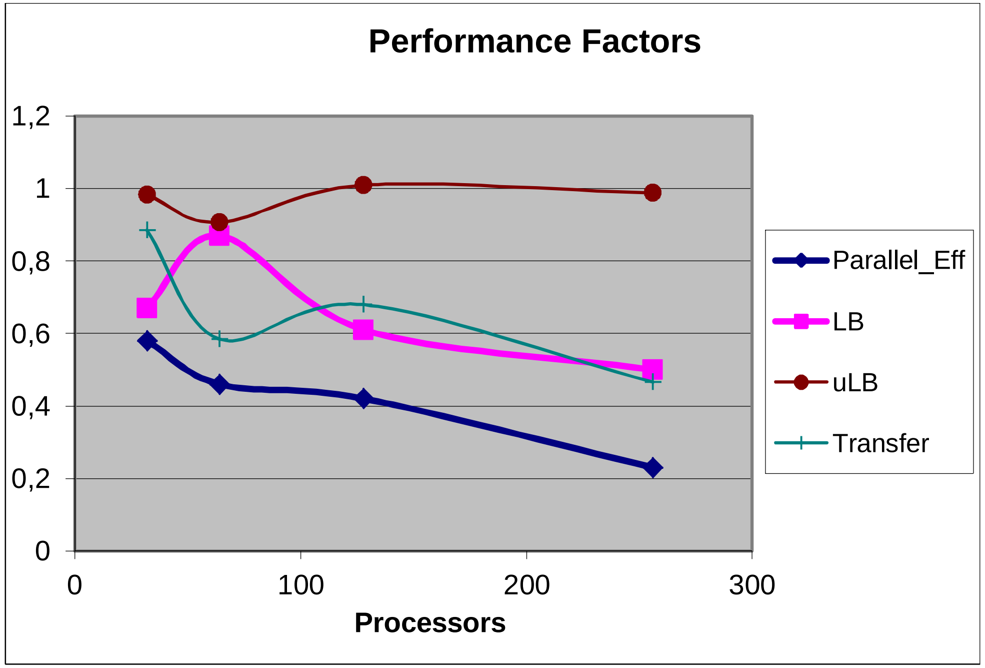
Platform	Processors	Input	Iteration time (s)	Speedup	Relative efficiency	Parallel Efficiency	LB	microLB	Transfer
MN	64	A	78,71	1,00	1,00	0,66	0,90	0,94	0,78
MN	128	A	44,22	1,78	0,89	0,58	0,97	0,92	0,65
MN	256	A	22,78	3,46	0,86	0,56	0.95	0,77	0,76
BGP	64	A	250,41	1,00	1,00	0,87	0,90	0,99	0,97
BGP	128	A	131,78	1,90	0,95	0,86	0,96	0,98	0,91
BGP	256	A	79,10	3,17	0,79	0,75	0,95	0,89	0,90
ICE	64	A	40,41	1,00	1,00	0,88	0,91	0,97	0,83
ICE	128	A	21,65	1,87	0,93	0,68	0,98	0,95	0,73
ICE	256	A	12,15	3,32	0,83	0,61	0,94	0,95	0,68

Although we have observed production runs at BSC with horrible load balance

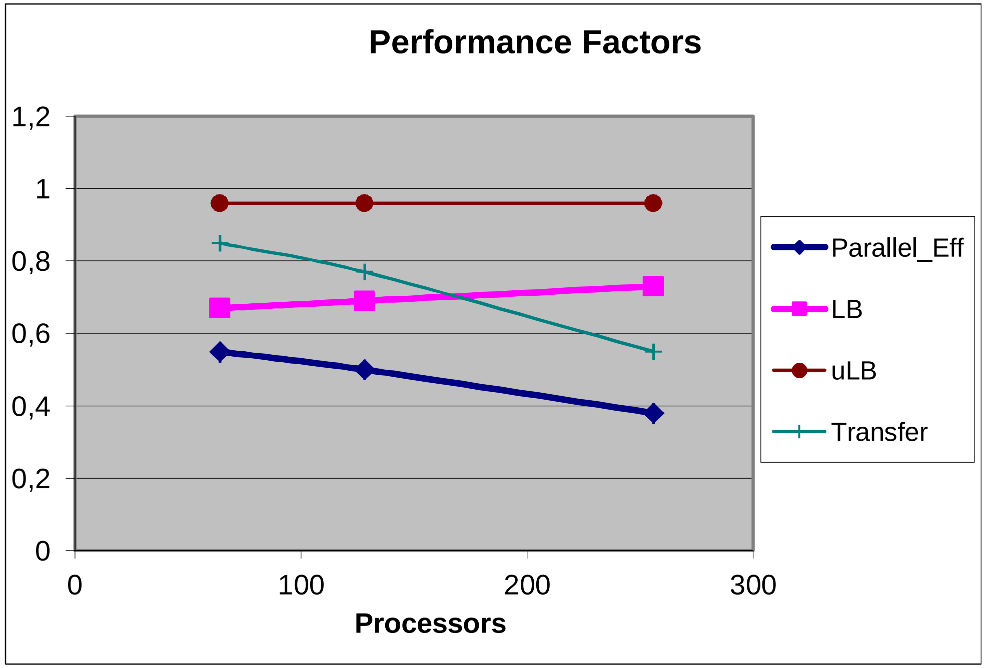
Parallel Performance Models

GROMACS

Original Version



v4.5



Old → new version: -43% instr, -52% time

Outline

Extrae
Paraver
Dimemas

Scaling model
Structure detection
HWC analyses
  Projection and CPI Stack models
  Folding

Scalability

Performance @ serial computation bursts

Burst = continuous computation region

between exit of an MPI call and entry to the next

Scatter plot representation of bursts

Collapse time dimension

N dimensional space of HWC derived relevant metrics

- Instructions: idea of computational complexity, computational load imbalance,…
- IPC: Idea of absolute performance and performance imbalance

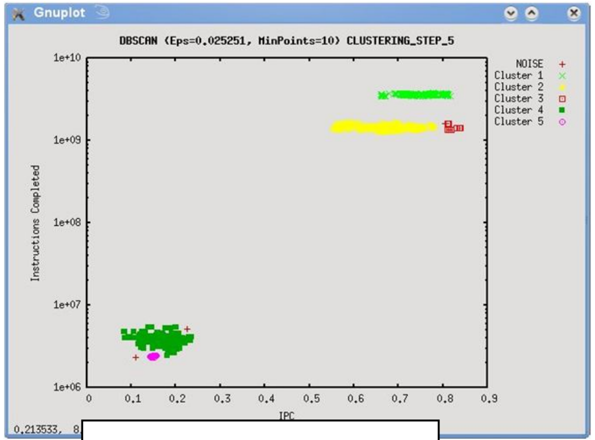
Structure

Clouds, clusters: Burst of “similar” characteristics

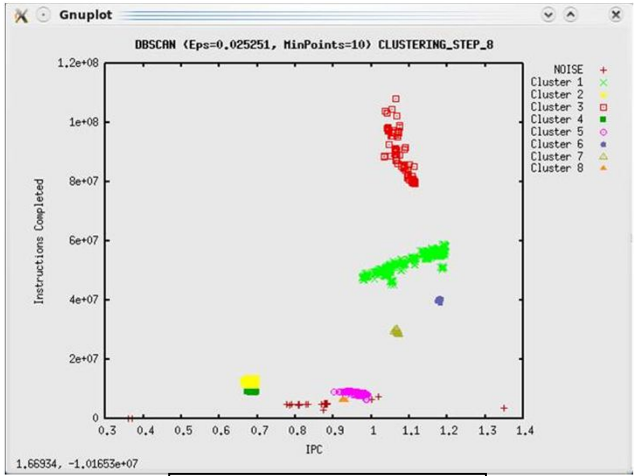
How those similarities spread in the timeline?

How does it relate to source code?

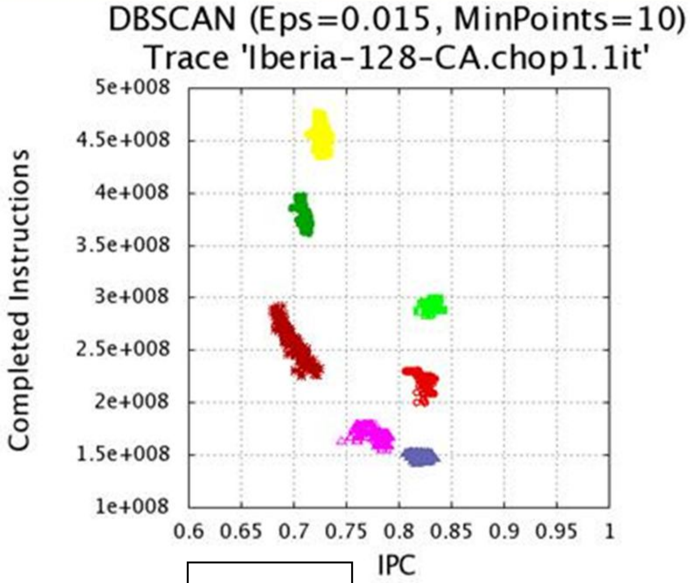
Performance @ serial computation bursts



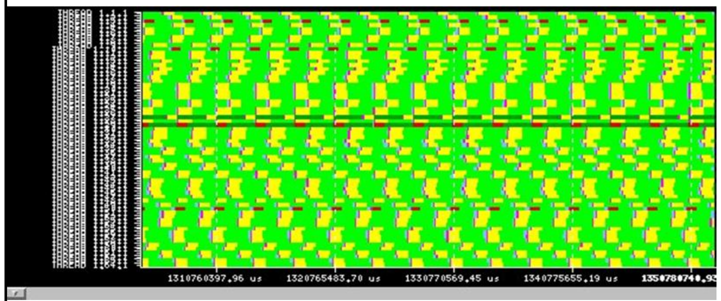
SPECFEM3D

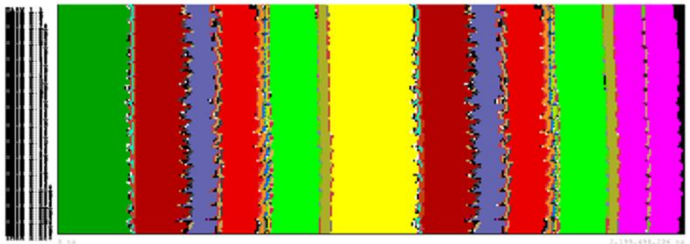


GROMACS



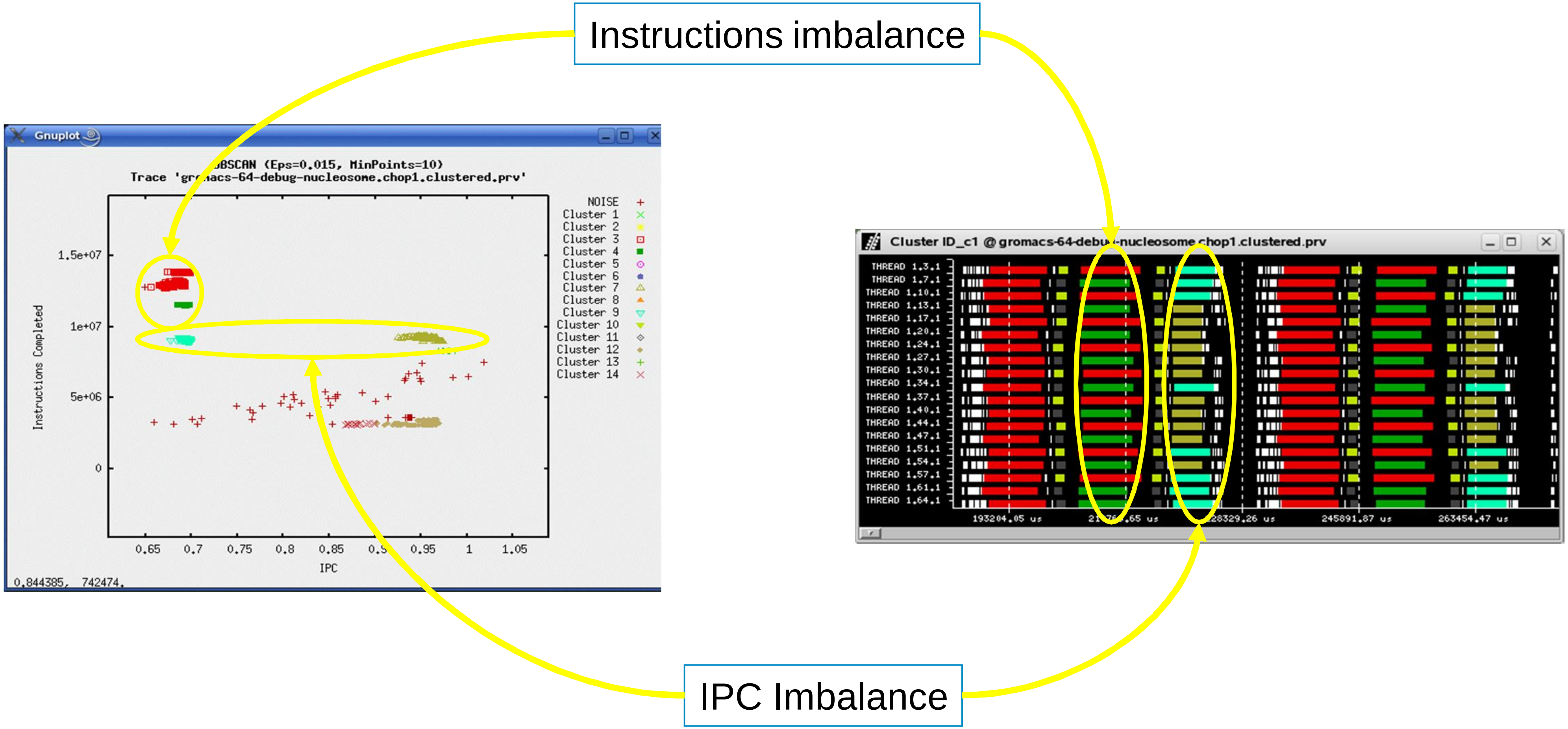
WRF





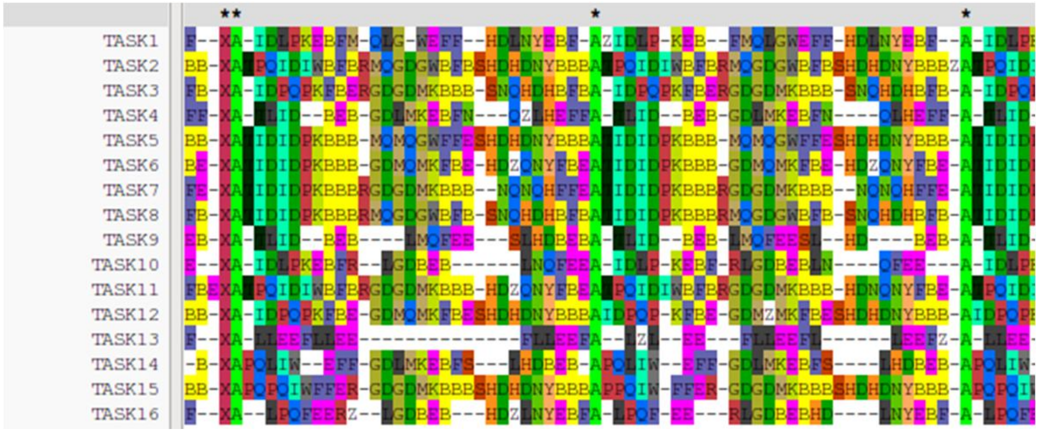
The importance of joint timeline+scatterplots

GROMACS FFTs balance

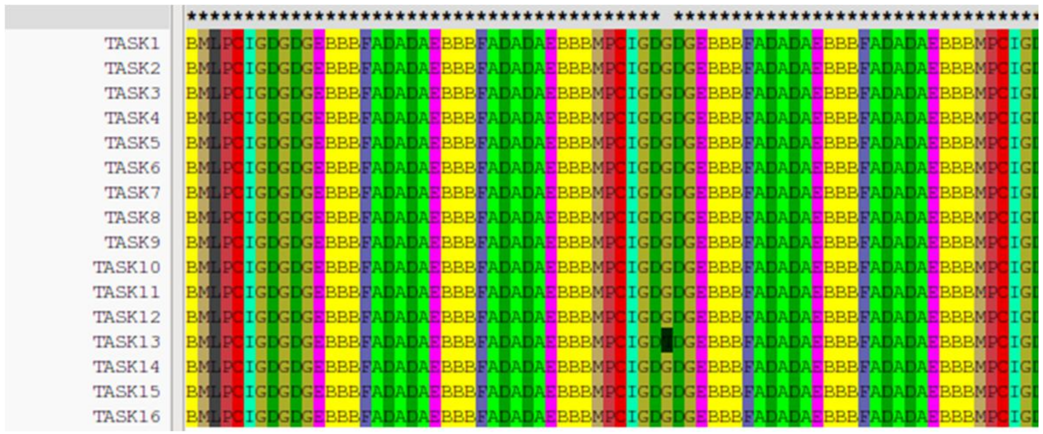


Automatic clustering quality assessment

- Leverage Multiple Sequence alignment tools from Life Sciences
- Process == Sequence of clusters ↔ sequence of amino acids == DNA
- CLUSTAL W, T-Coffee, Kalign2
- Cluster Sequence Score (0..1)
- Per cluster / Global
  - Weighted average



BT.A 0022



BT.A 0043

Outline

- Extrae
- Paraver
- Dimemas

- Scaling model
- Structure detection
- HWC analyses
  - Projection and CPI Stack models
  - Folding

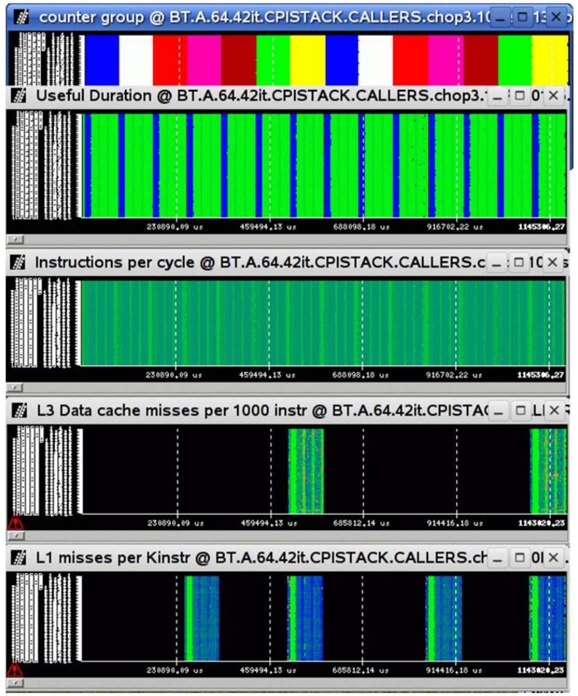
- Scalability

HWC projection

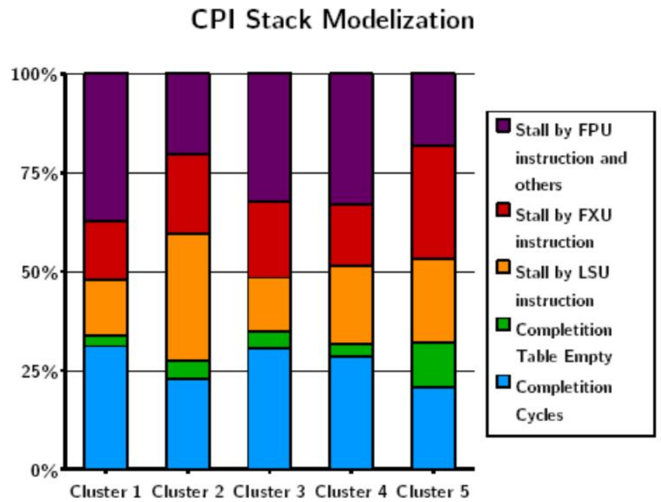
Full characterization

- All HWCs and ratios between them
- CPI stack model

From a single run



Cluster	1	2	3	4	5
%Time	54.88	17.96	16.90	6.44	1.42
Avg. Burst Dur. (ms)	1.02	0.78	13.14	2.50	1.11
IPC	1.02	0.65	0.89	0.91	0.53
MIPS	2231.8	1423.3	1966.5	2001.8	1163.0
MFLOPS	339.2	46.3	191.6	269.2	23.6
L1M/KInstr	0.92	1.53	1.19	1.17	2.88
L2M/KInstr	0.06	1.26	0.06	0.35	0.21
Mem.BW (MB/s)	16.79	218.47	13.87	85.77	29.76



Outline

Extrae
Paraver
Dimemas

Scaling model
Structure detection
HWC analyses
  Projection and CPI Stack models
  Folding

Scalability

Folding: Instrumentation + sampling

Extremely detailed time evolution of hardware counts, rates and callstack

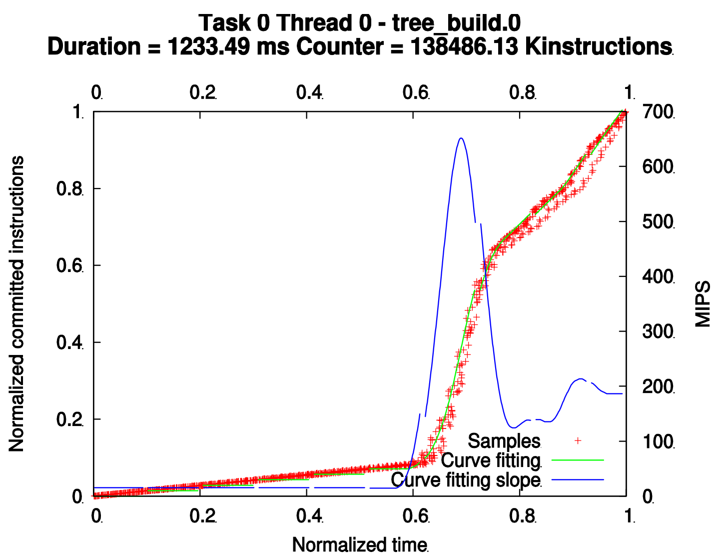
Minimal overhead

Based on

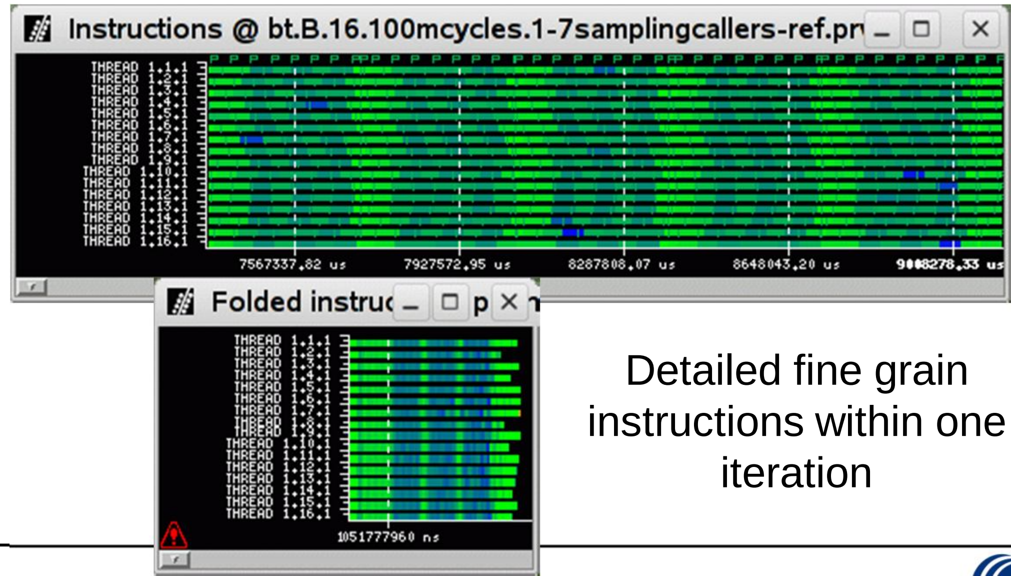
- trace: instrumentation events (iteration, MPI, …) and periodic samples.
- Application structure: manual iteration instrumentation, routines, clusters

Folding

- Postprocessing to project all samples into one instance



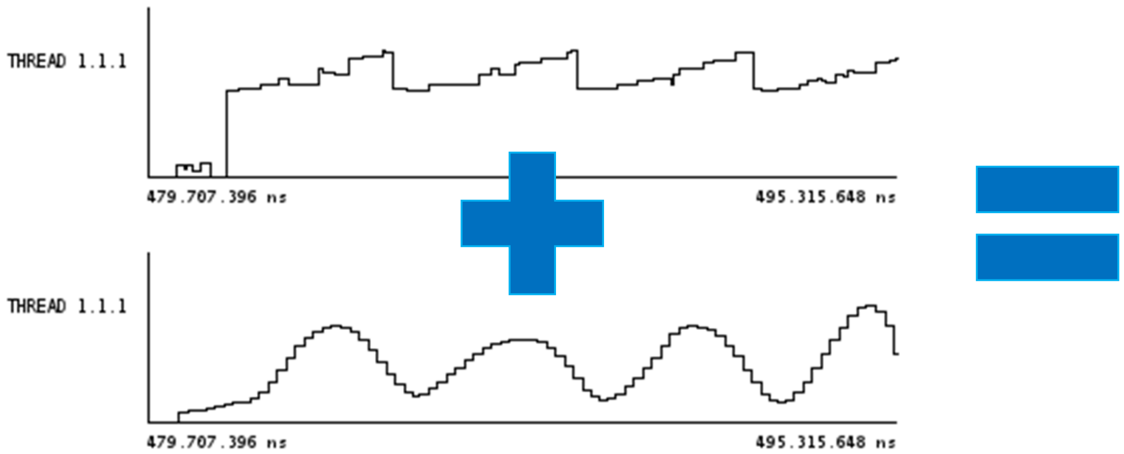
Original sampled instructions



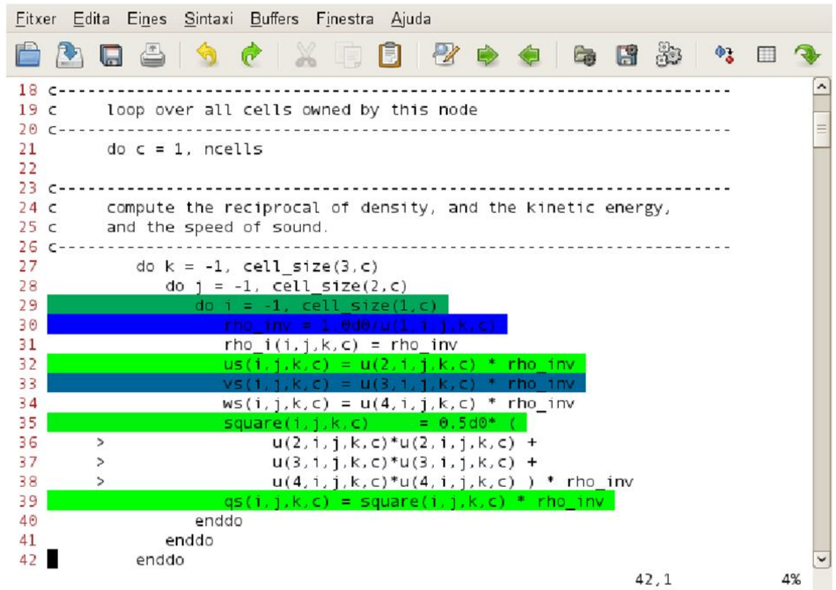
Detailed fine grain instructions within one iteration

Folding → profiles of rates

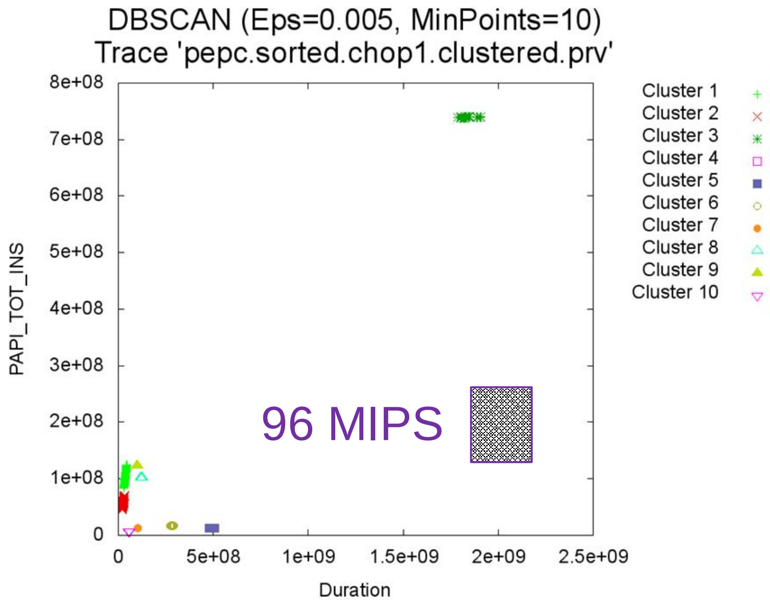
Folded source code line

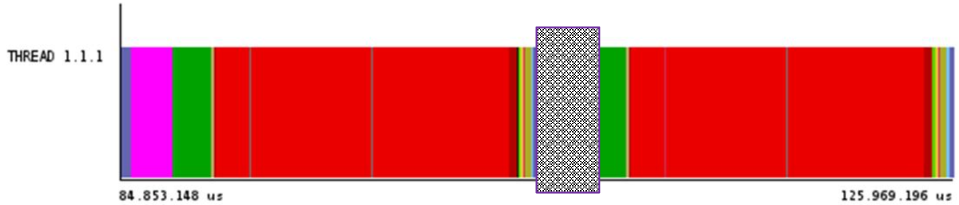


Folded instructions

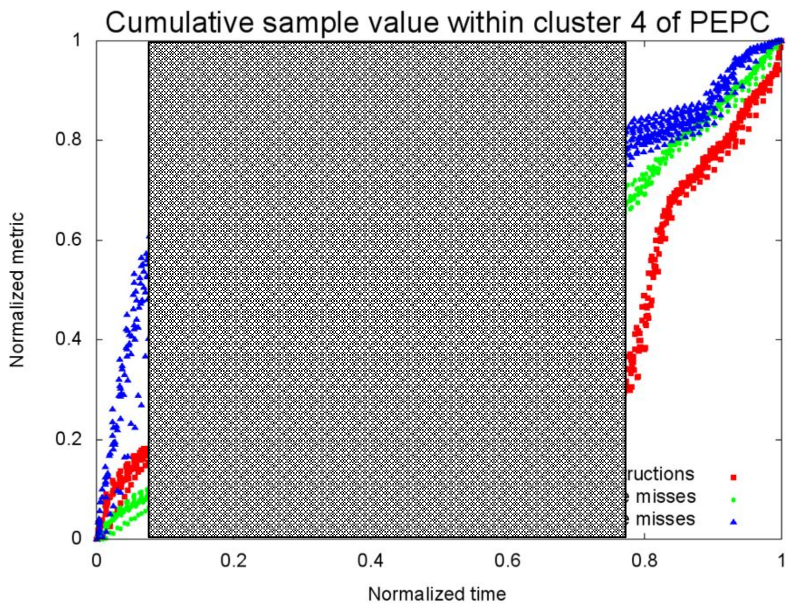


PEPC





Performance metrics
16 MIPS
2.3 M L2 misses/s
0.1 M TLB misses/s




```
htable%node = 0
htable%key = 0
htable%link = -1
htable%leaves = 0
htable%childcode = 0
```

→

```
do i = 1, n
 htable(i)%node = 0
 htable(i)%key = 0
 htable(i)%link = -1
 htable(i)%leaves = 0
 htable(i)%childcode = 0
End do
```

Changes
-70% time
-18% instructions
-63% L2 misses
-78% TLB misses
253 MIPS (+163%)

Outline

Extrae
Paraver
Dimemas

Scaling model
Structure detection
HWC analyses
- Projection and CPI Stack models
- Folding

Scalability

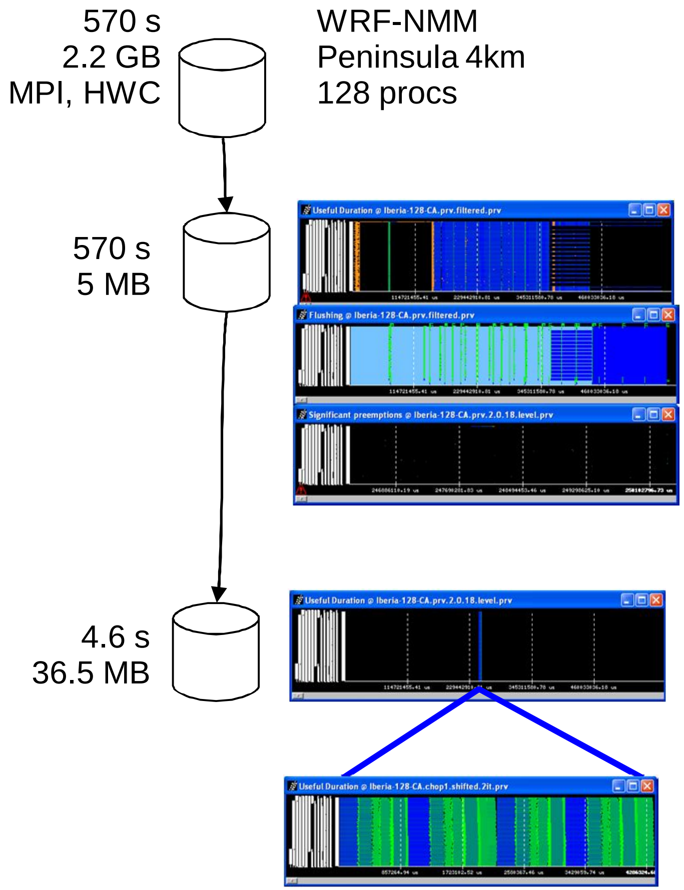
Data reduction

Data handling/summarization capability

- Software counters, filtering and cutting

Automatizable through signal processing techniques:

- Mathematical morphology to clean up perturbed regions
- Wavelet transform to identify coarse regions
- Spectral analysis for detailed periodic pattern



Algorithms applied to traces and online

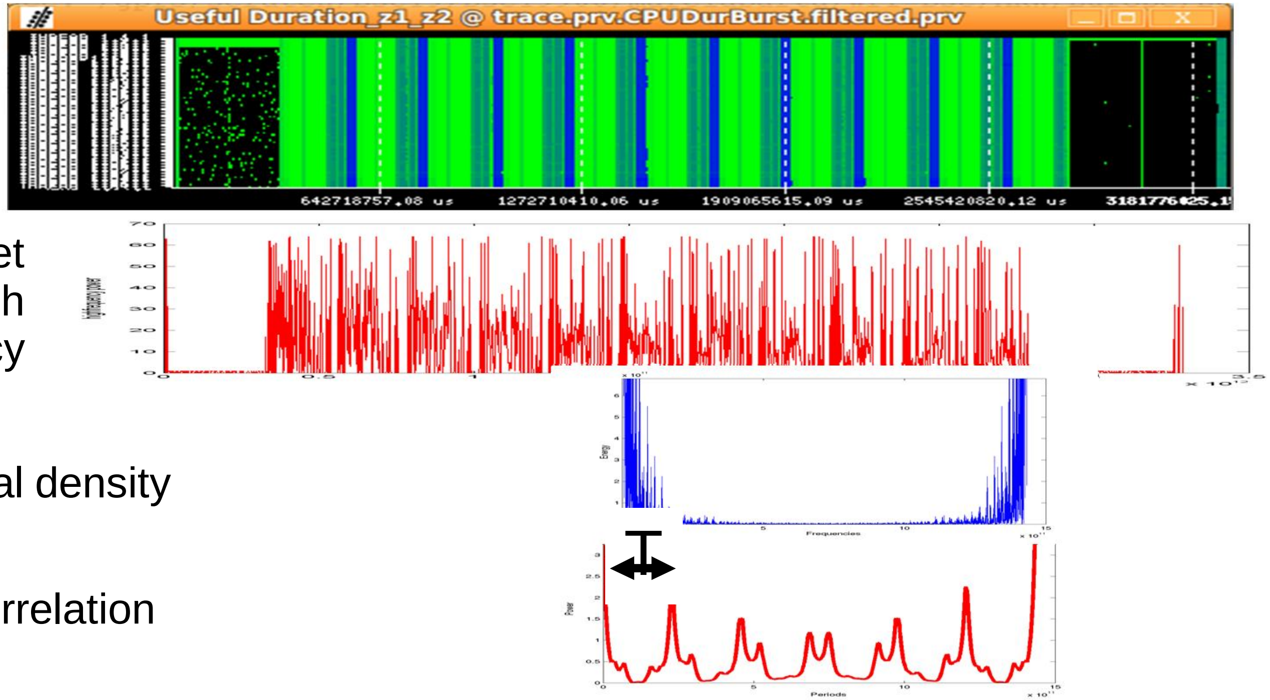
- Extrae (Stand alone or using on MRNET)
- Paraver

Spectral analysis

Wavelet
High
frequency

Spectral density

Autocorrelation



Scalability: online data reduction

Jugene

Jaguar

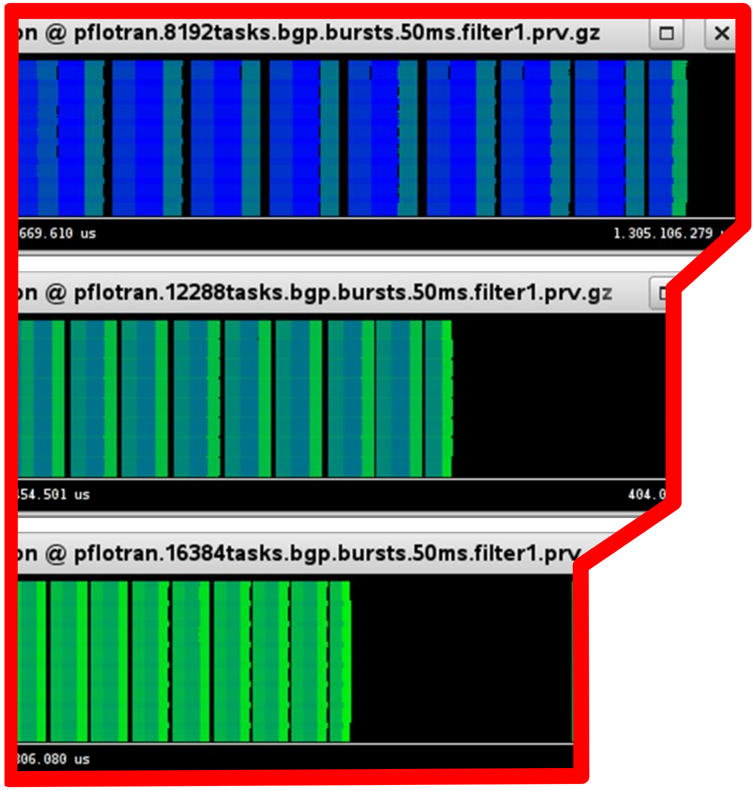
~ 105 seconds

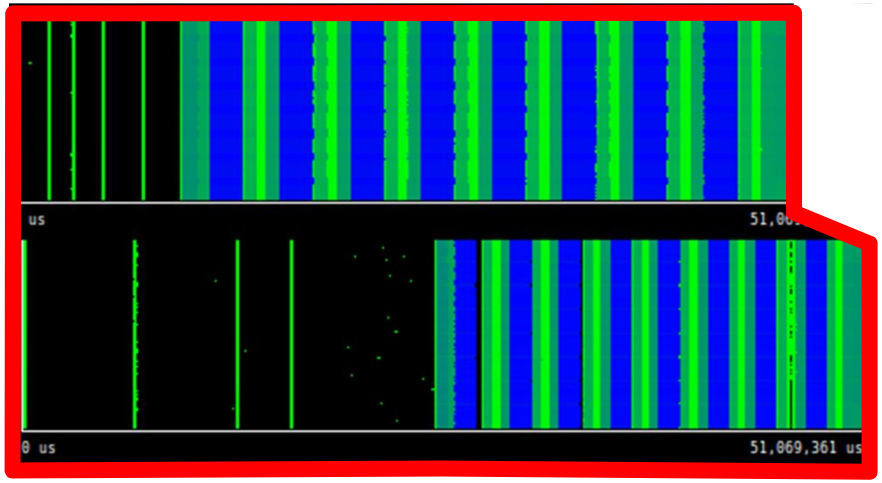
~ 47 seconds

8K cores

12K cores

16K cores





Flow

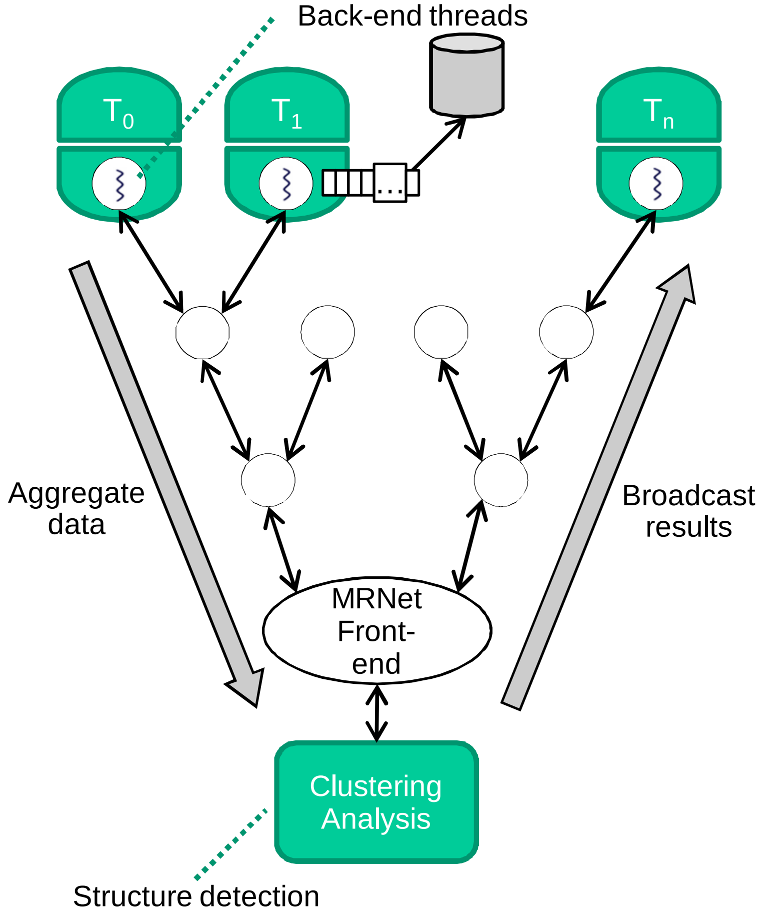
Tran

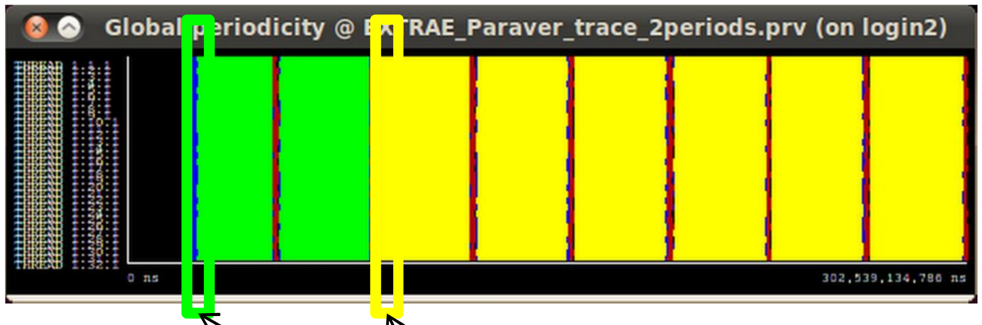
Tran

Flow

PFLOTRAN

Scalability: online automatic interval selection





Detait trace only for small interval

" G. Llort et all, "Scalable tracing with dynamic levels of detail" ICPADS 2011

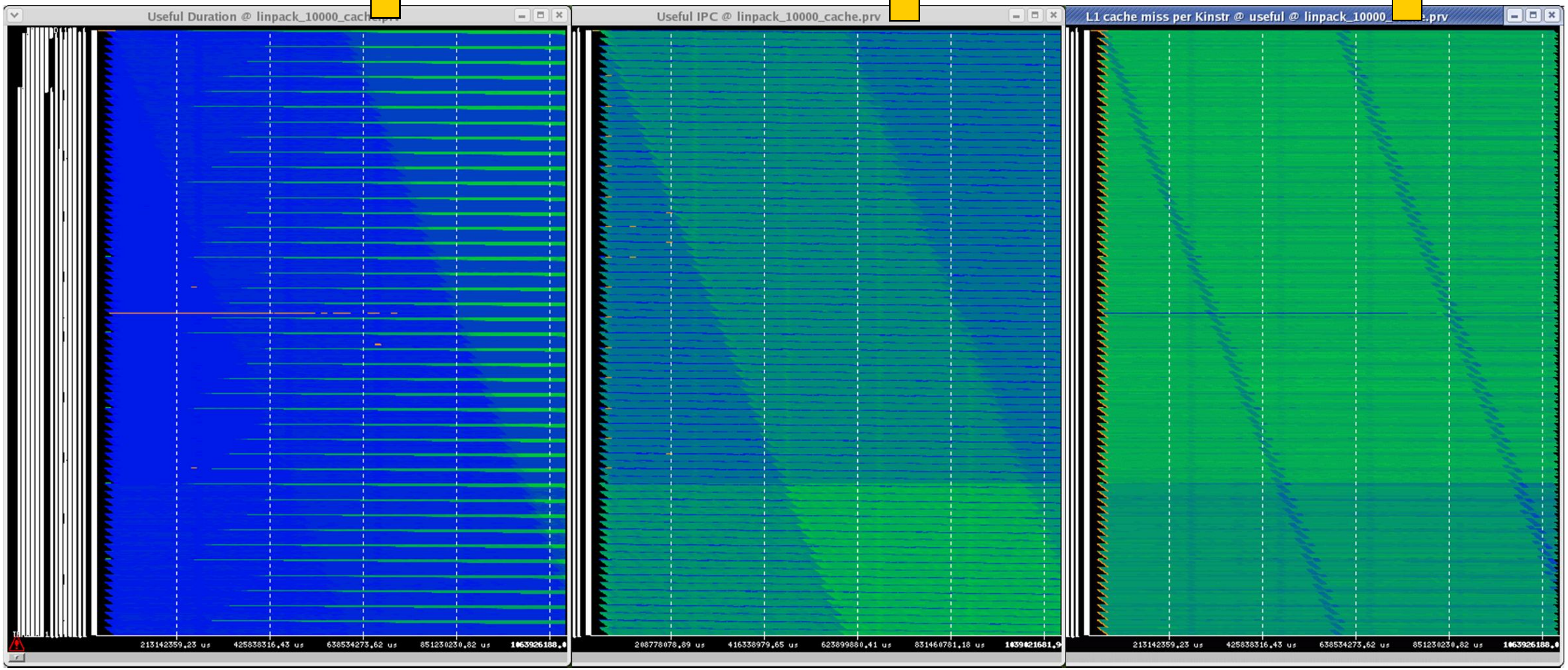
Scalable display: Non linear rendering

Linpack @ Marenostrum: 10k cores x 1700 s

Dgemm duration — 11.8 s / 10 s

Dgemm IPC — 2.95 / 2.85

Dgemm L1 miss ratio — 0.8 / 0.7



Conclusion

- Extreme flexibility:
  - Maximize iterations of the hypothesis – validation loop
  - Learning curve
  - “Don’t ask whether something can be done, ask how can it be done”
- Detailed and precise analysis
  - Squeeze the information obtainable form a single run
  - Insight and correct advise with estimates of potential gain
- Data analysis techniques applied to performance data

www.bsc.es/paraver