www.bsc.es

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

PERFORMANCE ANALYSIS WITH BSC-TOOLS DIMEMAS SIMULATOR

Juan Gonzalez – juan.gonzalez@bsc.es

PATC Tools Training. Barcelona,
Oct. 14-18th 2013

Introduction

What?

- Coarse grain trace driven simulator
- Message-passing applications

Use?

- Developers → Understand applications
- System administrators → Evaluate hardware

Introduction (II)

Key factors

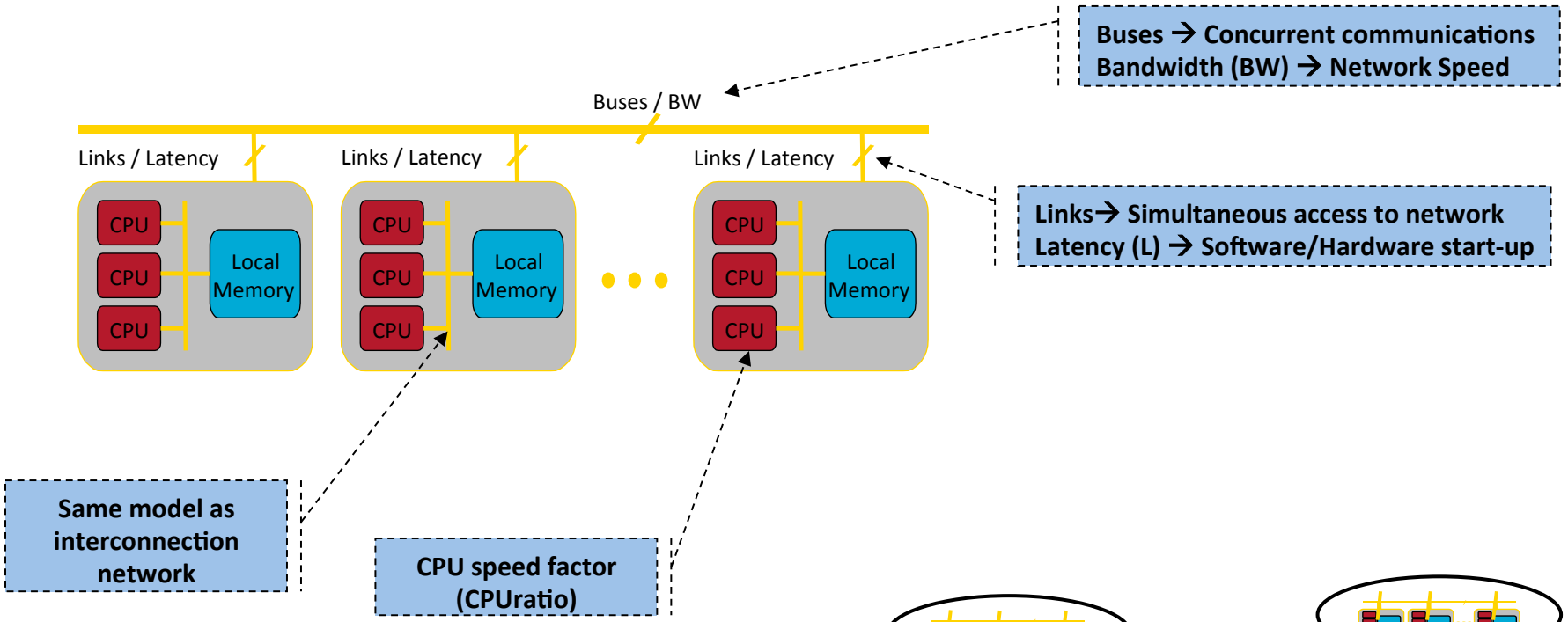
- Abstract architecture
  - Network of SMPS
- Basic MPI protocols
- No attempt to model details

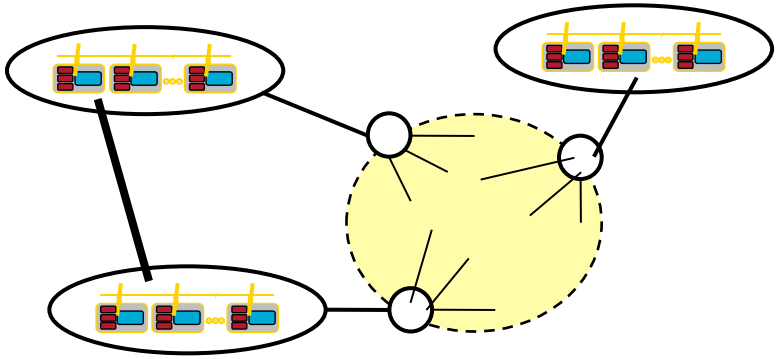
Objectives

- Simple / General
- Fast simulations

DIMEMAS MODEL

DIMEMAS simulated architecture





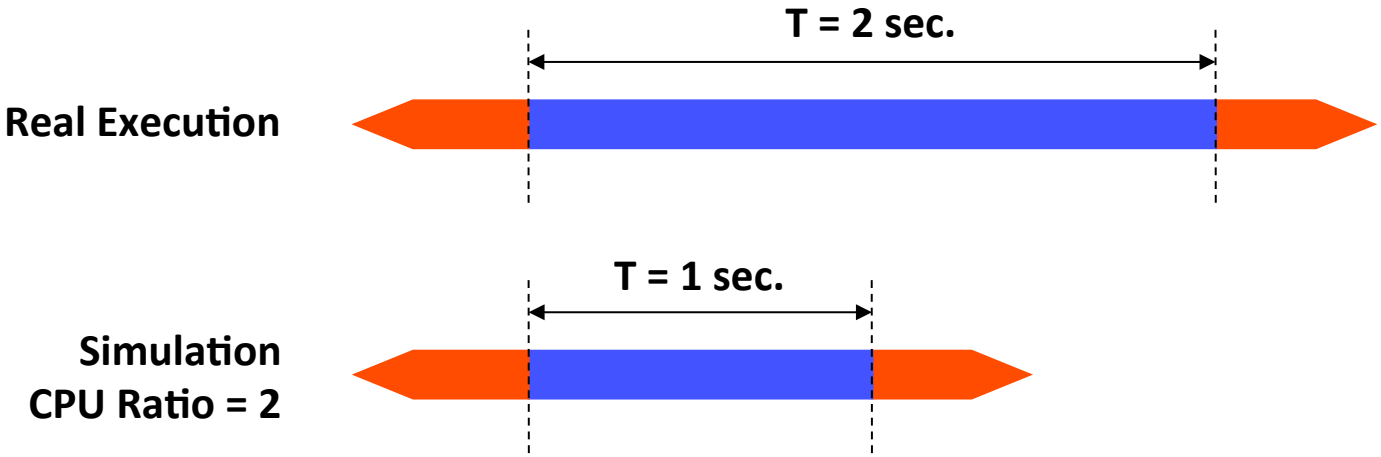
DIMEMAS Primitives: CPU bursts

What?

- Sequential computations

How?

- Clock advance
- *CPU Ratio* divisive factor



DIMEMAS Primitives: p2p communications

What?

- Message transmission between 2 partners

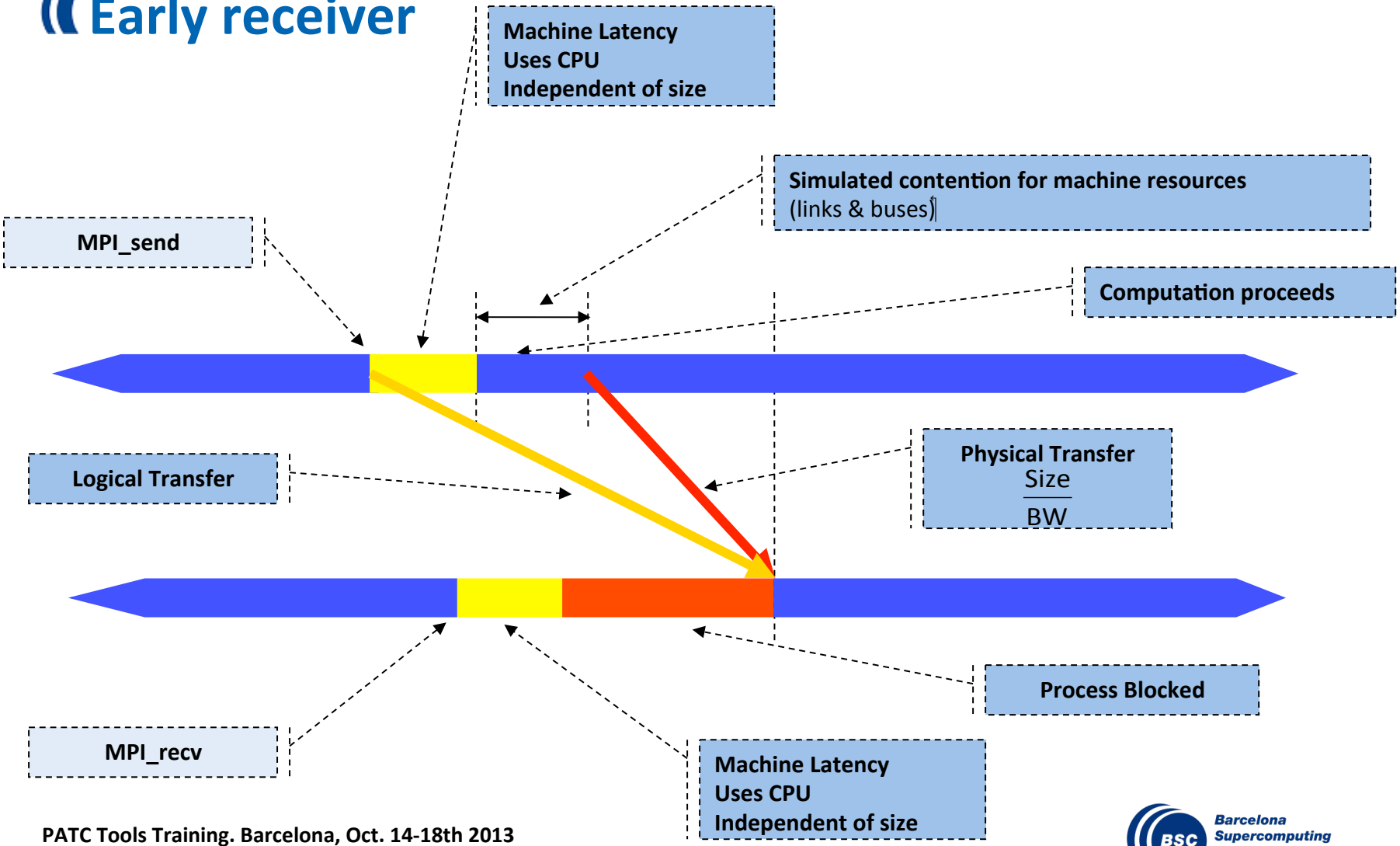
How?

- Resource allocation time (non-linear)
- Resource usage time (linear)

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

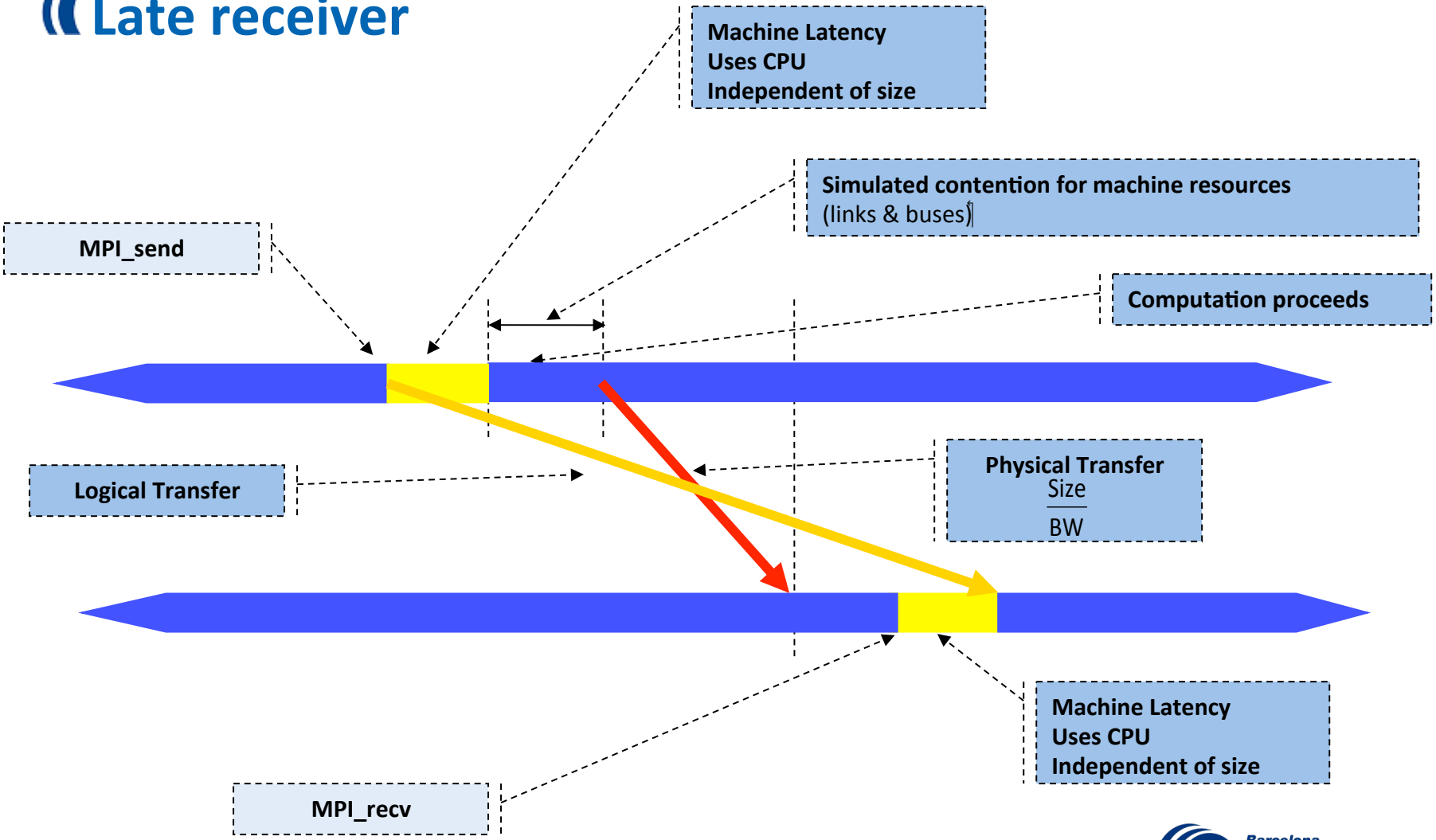
p2p communication model

Early receiver



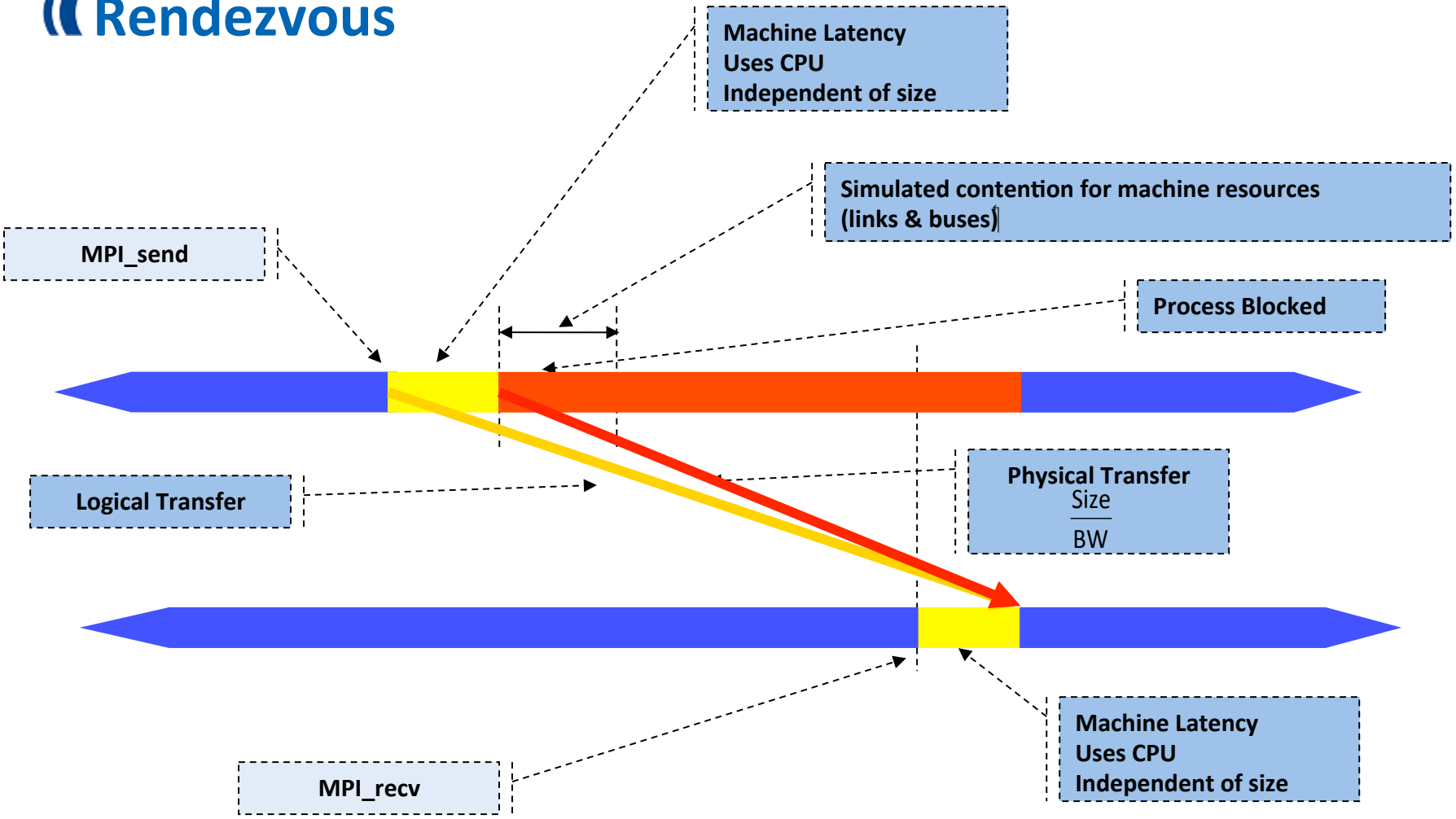
p2p communication model

Late receiver



p2p communication model

Rendezvous



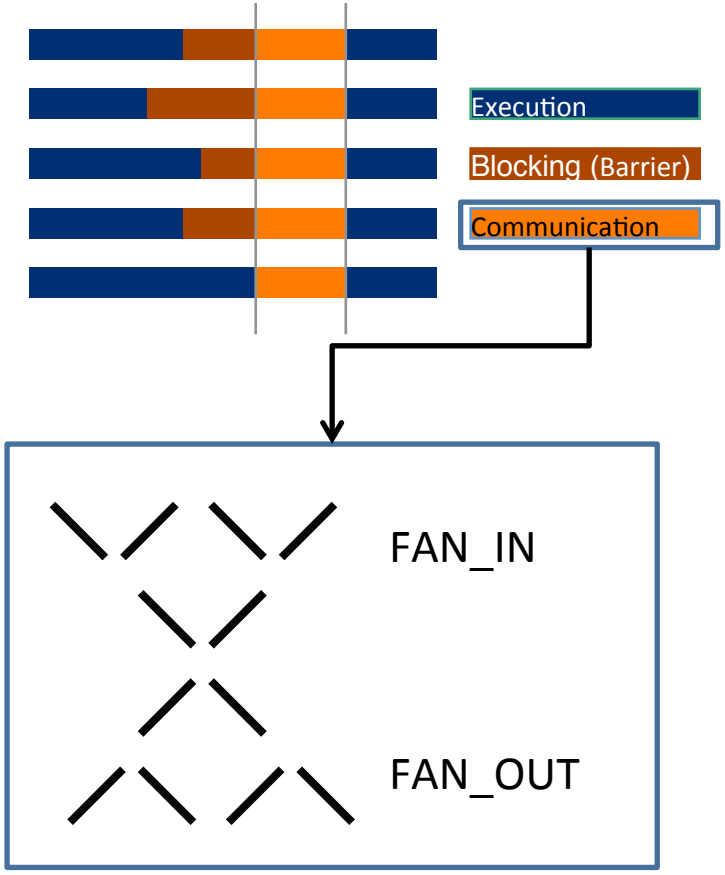
DIMEMAS Primitives: collective communications

What?

- Information transmission between multiple partners

How?

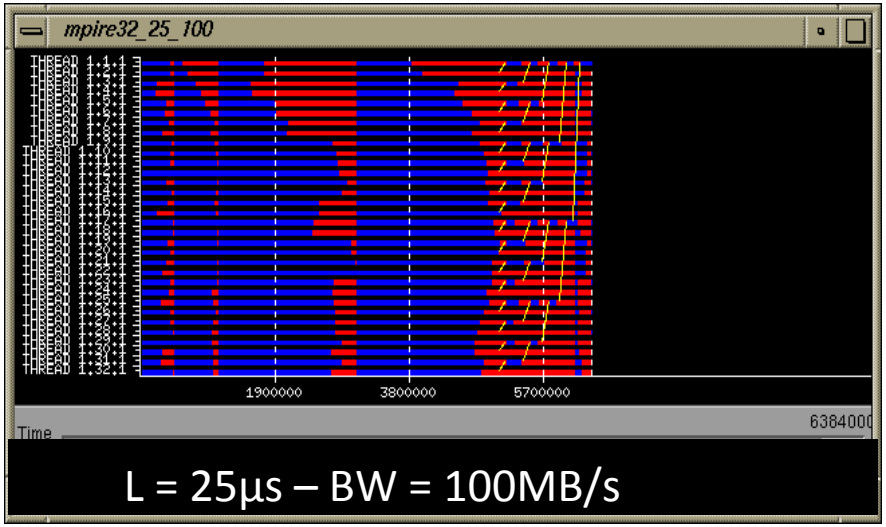
- Barrier: synchronization (non-linear)
- FAN_IN: information gathering
- FAN_OUT: information scattering
- Each FAN similar to p2p communication but taking into account # processes



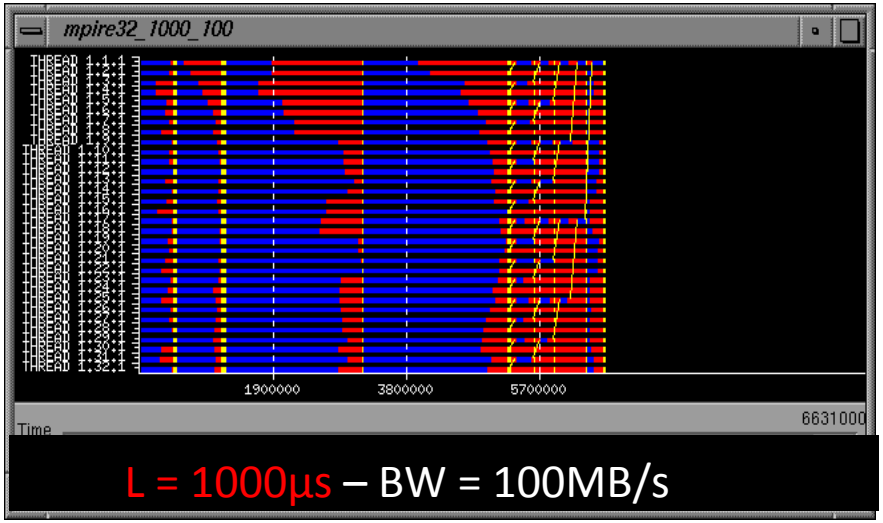
USE CASES

Network sensitivity (I)

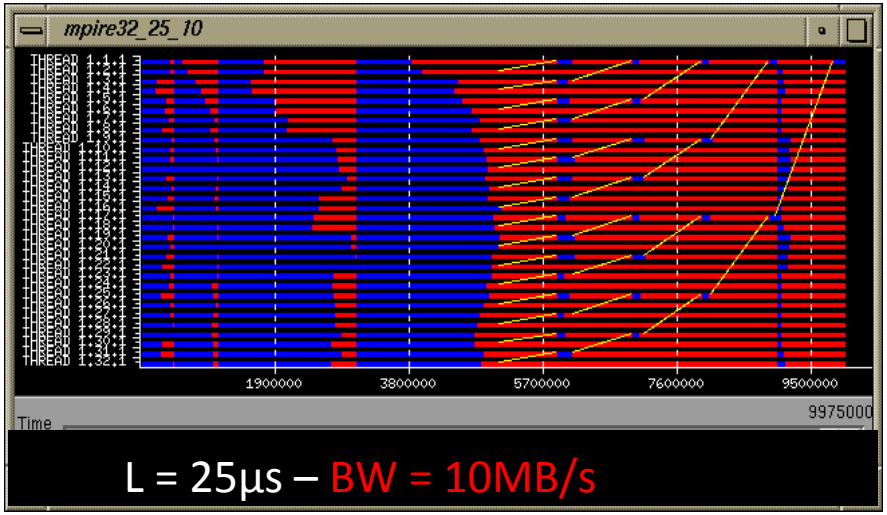
MPIRE 32 tasks, no network contention



L = 25µs – BW = 100MB/s



L = 1000µs – BW = 100MB/s



L = 25µs – BW = 10MB/s

All windows same scale

Network sensitivity (II)

WRF

- Iberia 4Km, 4procs/node
- NMM / ARW models
  - 128/256/512 procs. per model

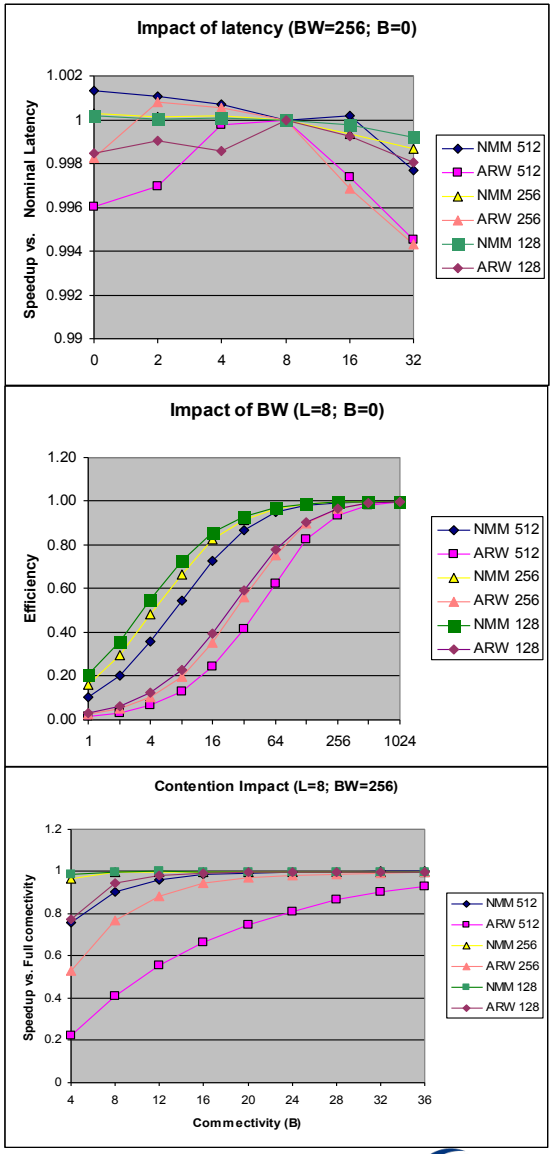
None sensitive to latency (L)

Bandwidth (BW)

- NMM: req. 256MB/s
- ARW: req. 1GB/s

Contention (B)

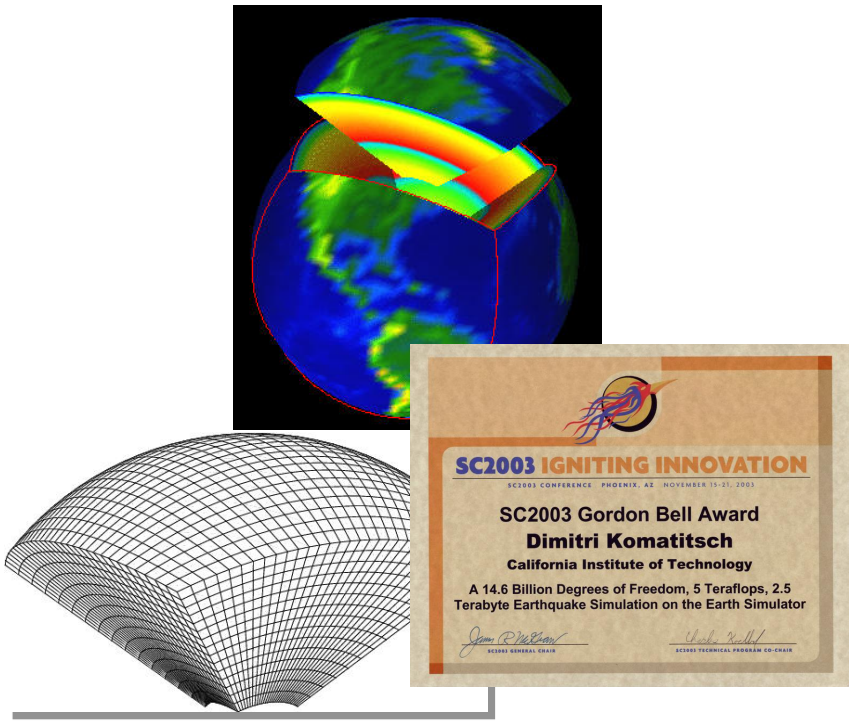
- NMM: sensitive at 512 procs. Scale
- ARW: sensitive at all scales



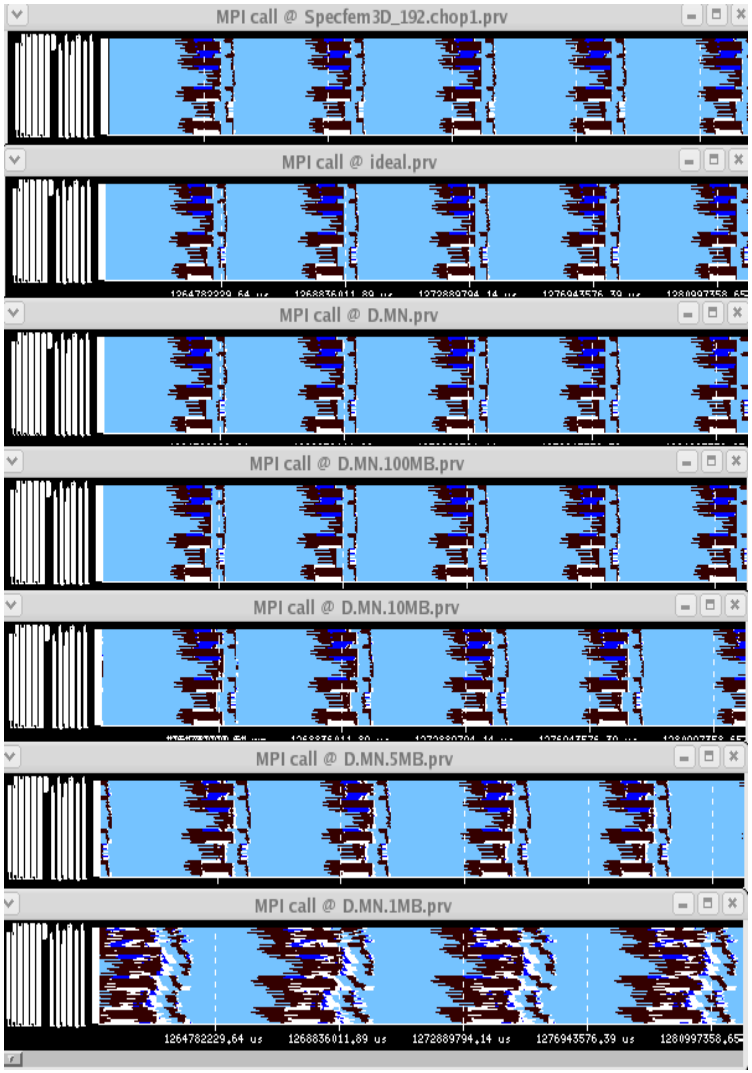
Performance prediction

SPECFEM3D

- Do we need asynchronous communications?



Courtesy Dimitri Komatitsch



Multi-scale simulation

Dimemas + CPU simulator

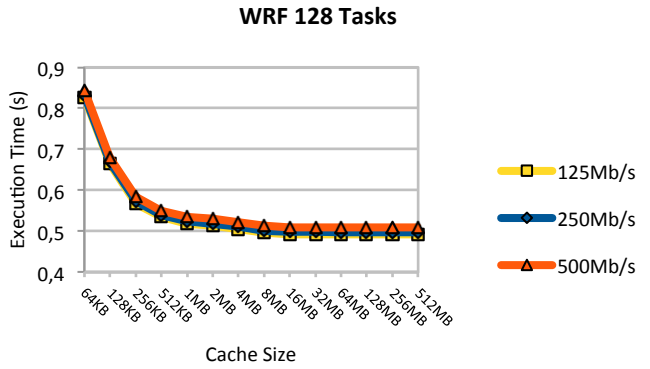
- Execution time using different cache size and network bandwidth

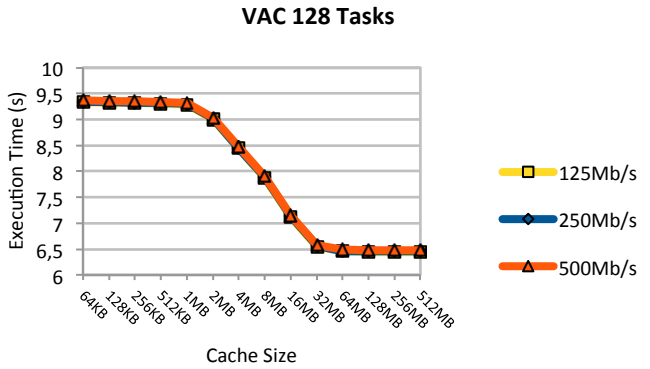
WRF & VAC

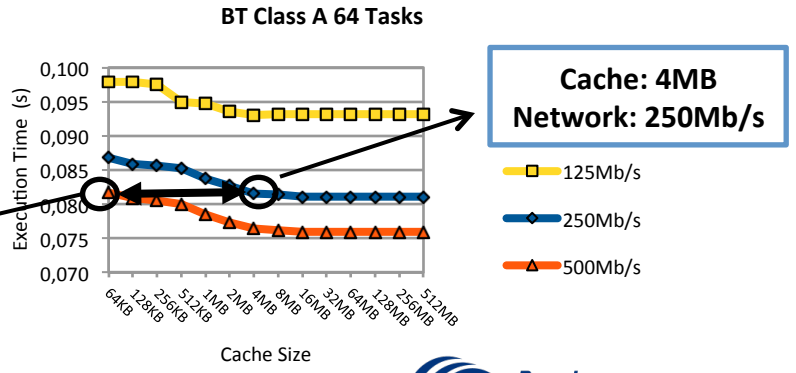
- Dominated by computation

NAS BT

- Can compensate cache reduction with more network BW



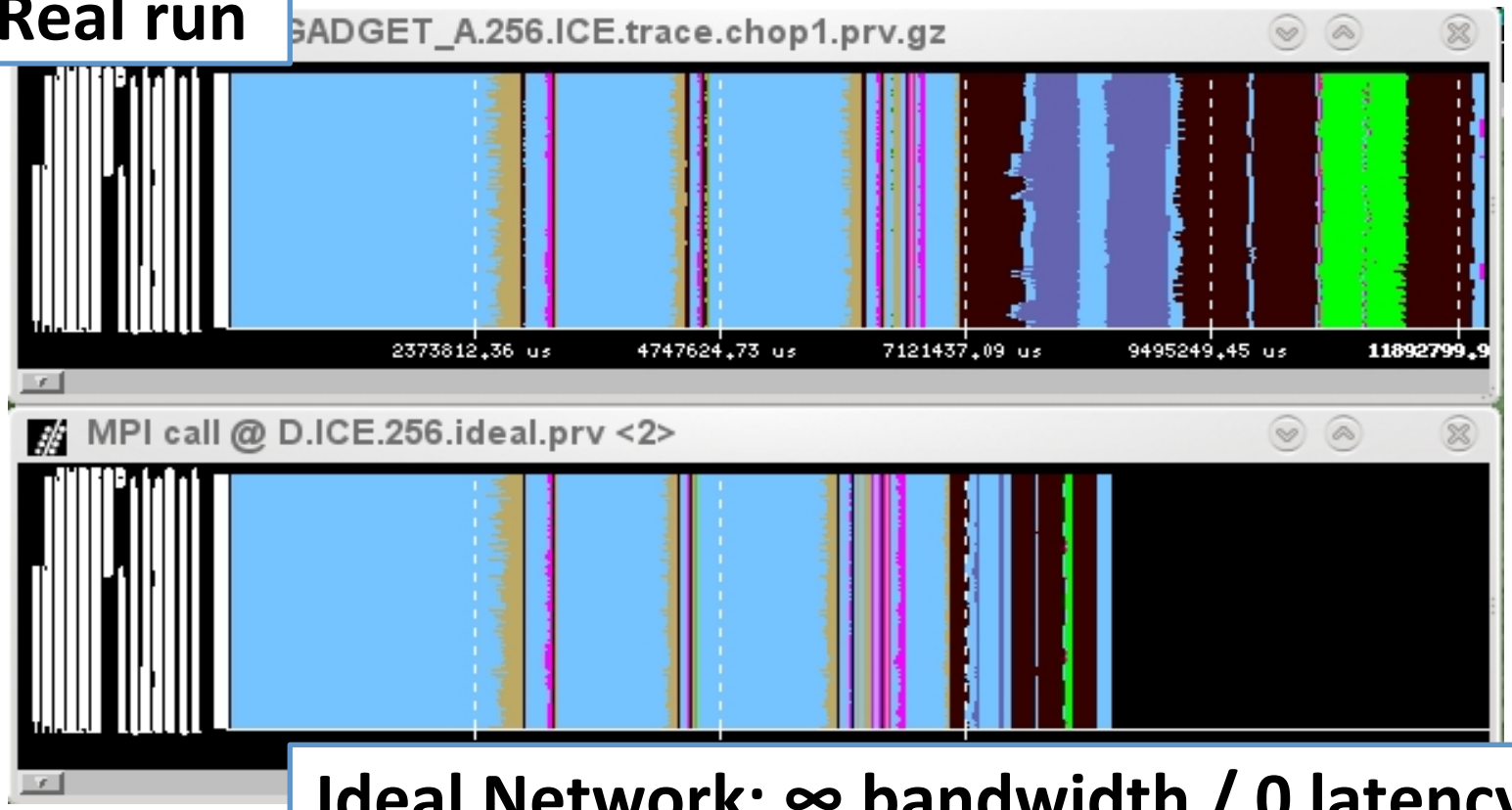




Application limits (I)

Load balance / dependences?

Real run

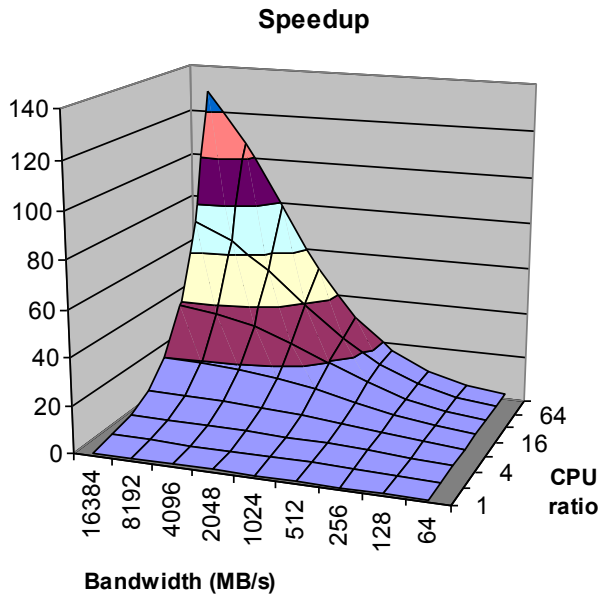


Ideal Network: ∞ bandwidth / 0 latency

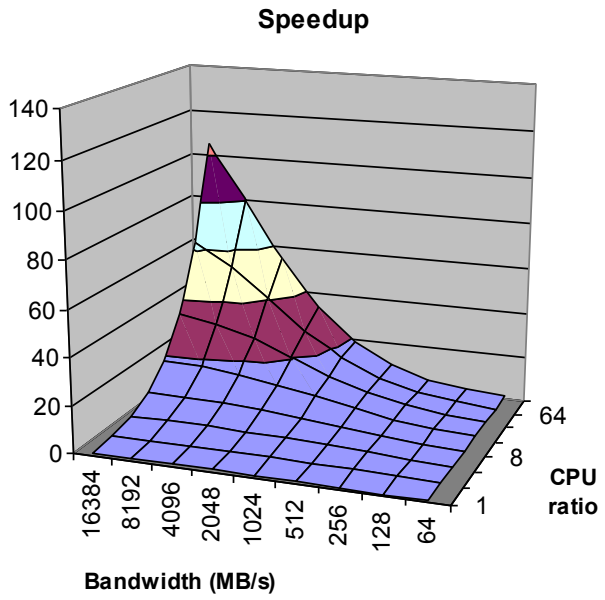
Application limits (II)

- Ideal speeding up ALL computation bursts by the CPUratio factor

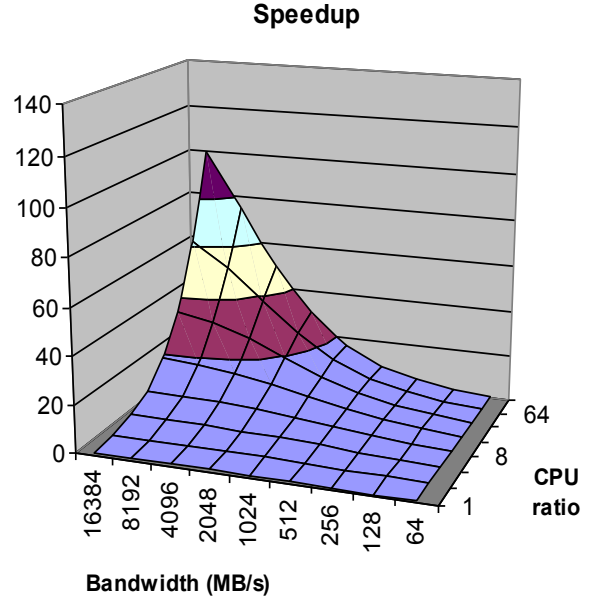
64 tasks



128 tasks



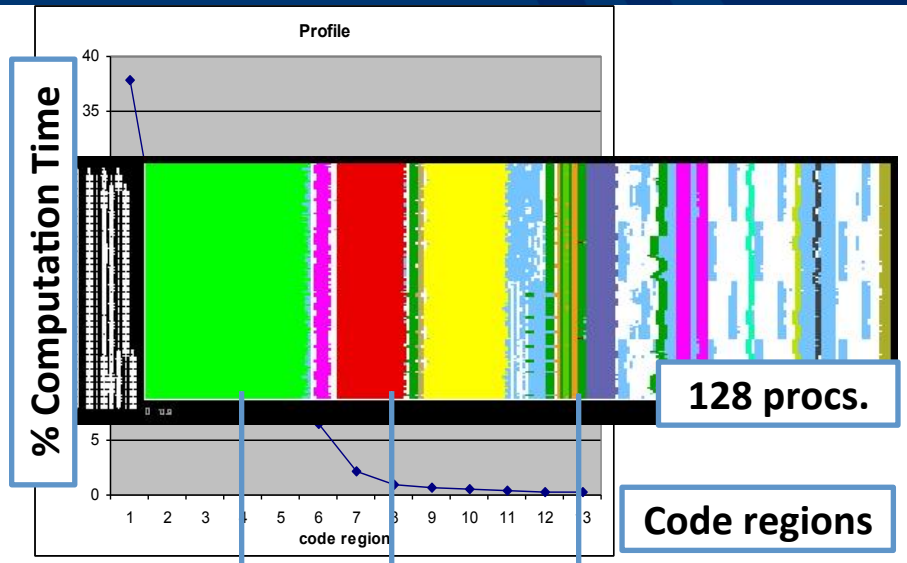
256 tasks

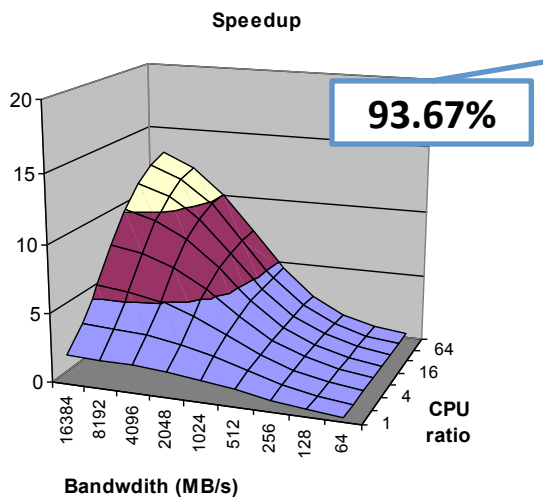


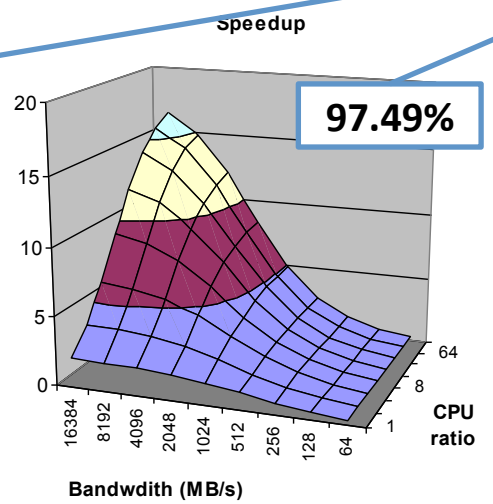
- The more processes, the less speed-up
  - Higher impact of bandwidth limitations!!!!

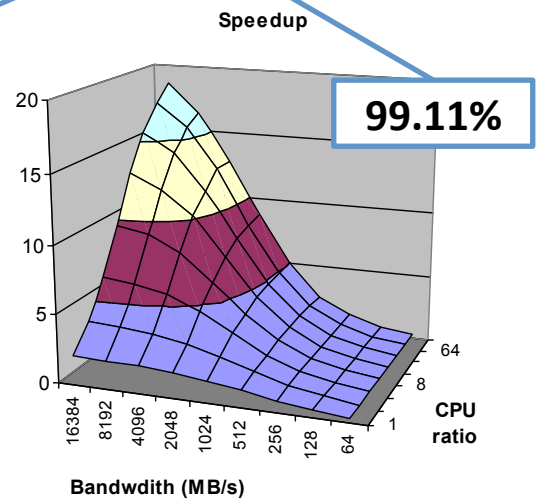
Application limits (III)

- Hybrid/accelerator parallelization
  - Speed-up SELECTED regions by the *CPUratio* factor
- We do need to overcome the hybrid Amdahl's law
  - Asynchrony + Load Balancing









(Previous slide: speed-ups **up to 120x**)

USE CASES

An in-depth study of the parallel efficiency…

Application's Efficiency (I)

CG-POP mpi2s1D 180x120 – 10 iterations

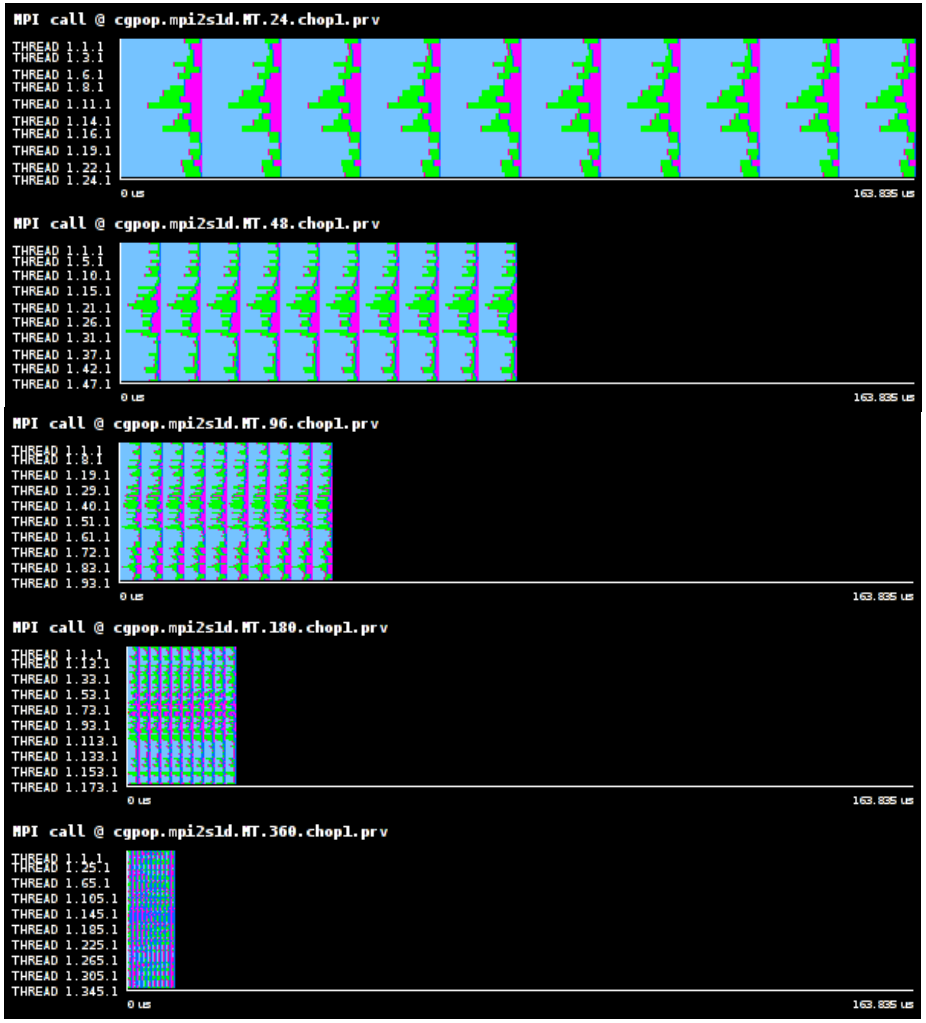
24 tasks

48 tasks

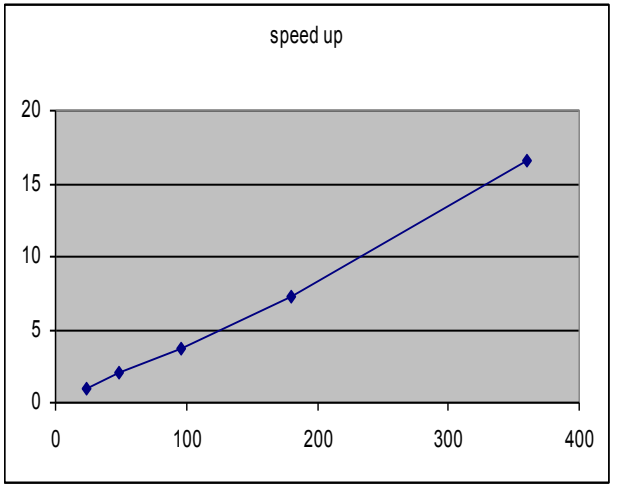
96 tasks

180 tasks

360 tasks

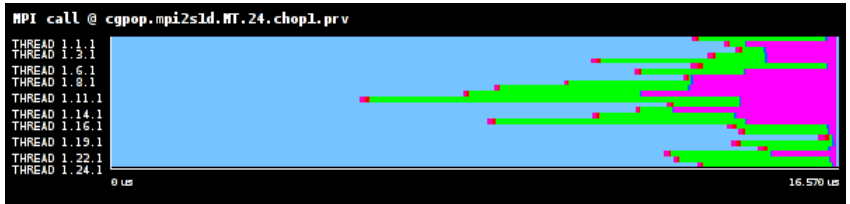
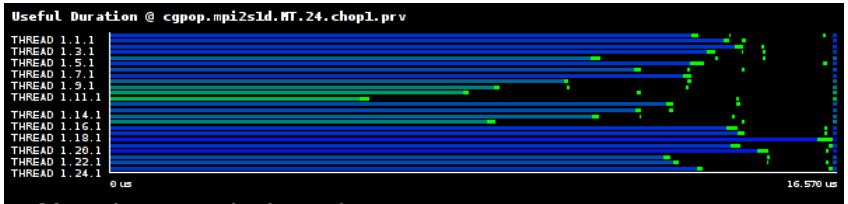
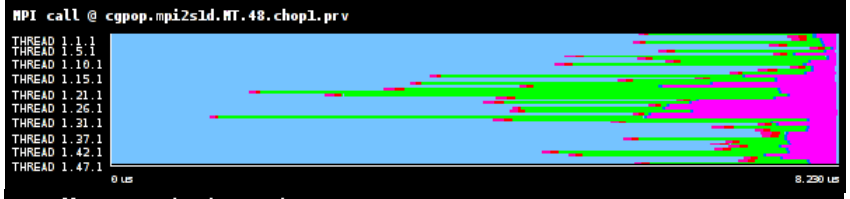
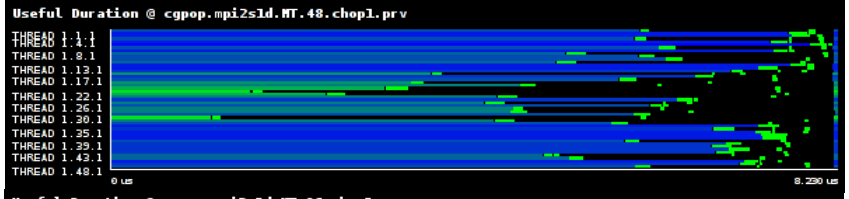
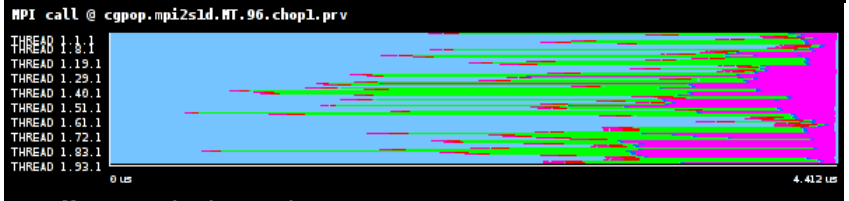
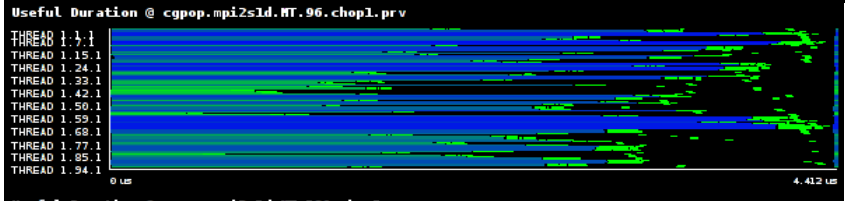
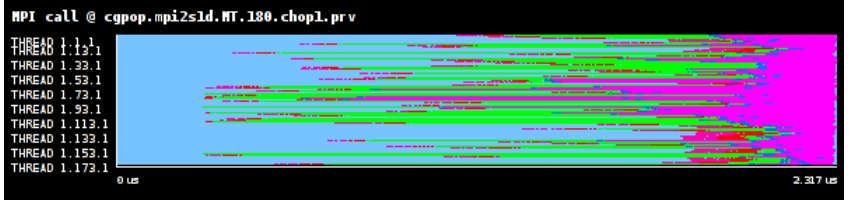
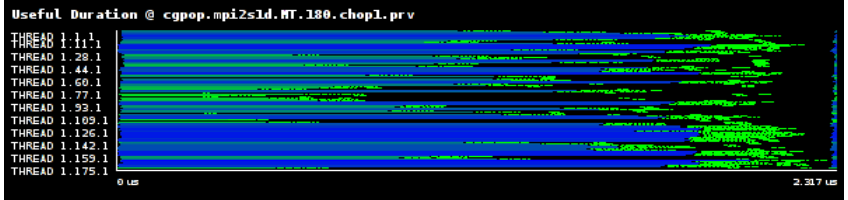
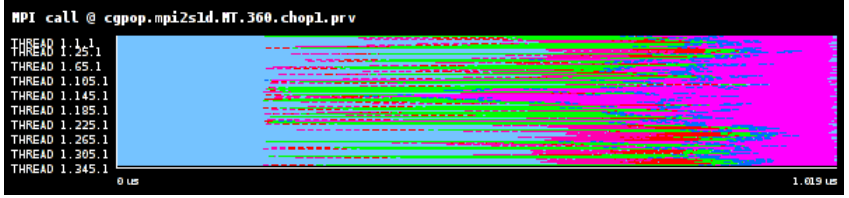
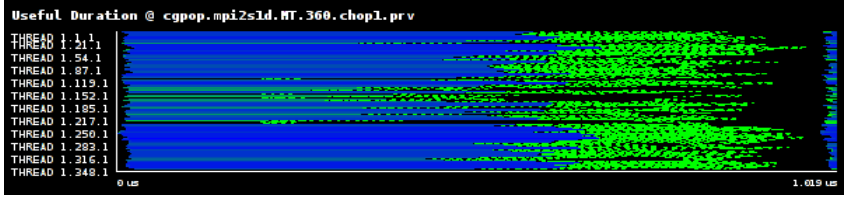


Good scalability!



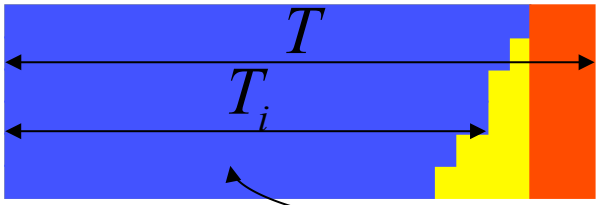
Application's Efficiency (II)

Comparing 1 iteration – different time scale

	MPI call view	Duration of computing phase
24		
48		
96		
180		
360		

Efficiency model: Parallel Efficiency

$$\eta = LB \times CommEff$$

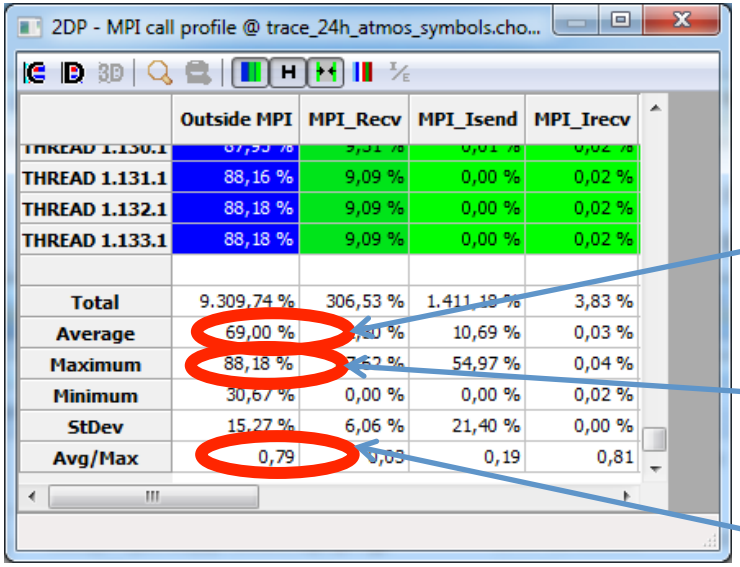


$$eff_i = \frac{T_i}{T}$$

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

IPC
#instr

Obtained from real execution metrics



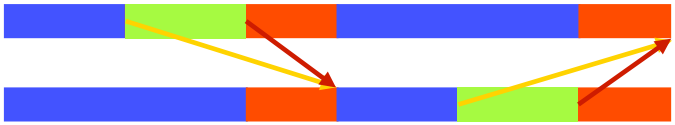
η

CommEff

LB

Efficiency model (II): µLB & transfer

But…



$$LB = 1$$

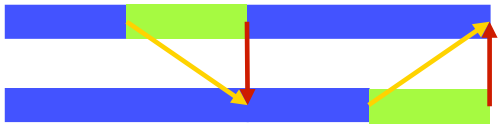
… there is other type of LB! µLB

- **Serializations / Dependences**

$$CommEff = \mu LB \times Transfer$$

We measure it using Dimemas!

- **How? Using an ideal network → Transfer efficiency = 1**



Efficiency model (III): Speedup model

Final model

$$S_P = \frac{P}{P_0} \times \frac{LB}{LB_0} \times \frac{\mu LB}{\mu LB_0} \times \frac{Transfer}{Trasnfer_0} \times \frac{IPC}{IPC_0} \times \frac{\#instr_0}{\#instr}$$

Parallel Efficiency

Computation Efficiency

M. Casas et. Al., *Automatic analysis of speedup of MPI applications*. ICS 2008

Application's Efficiency (I)

Comparing 10 iterations – same time scale

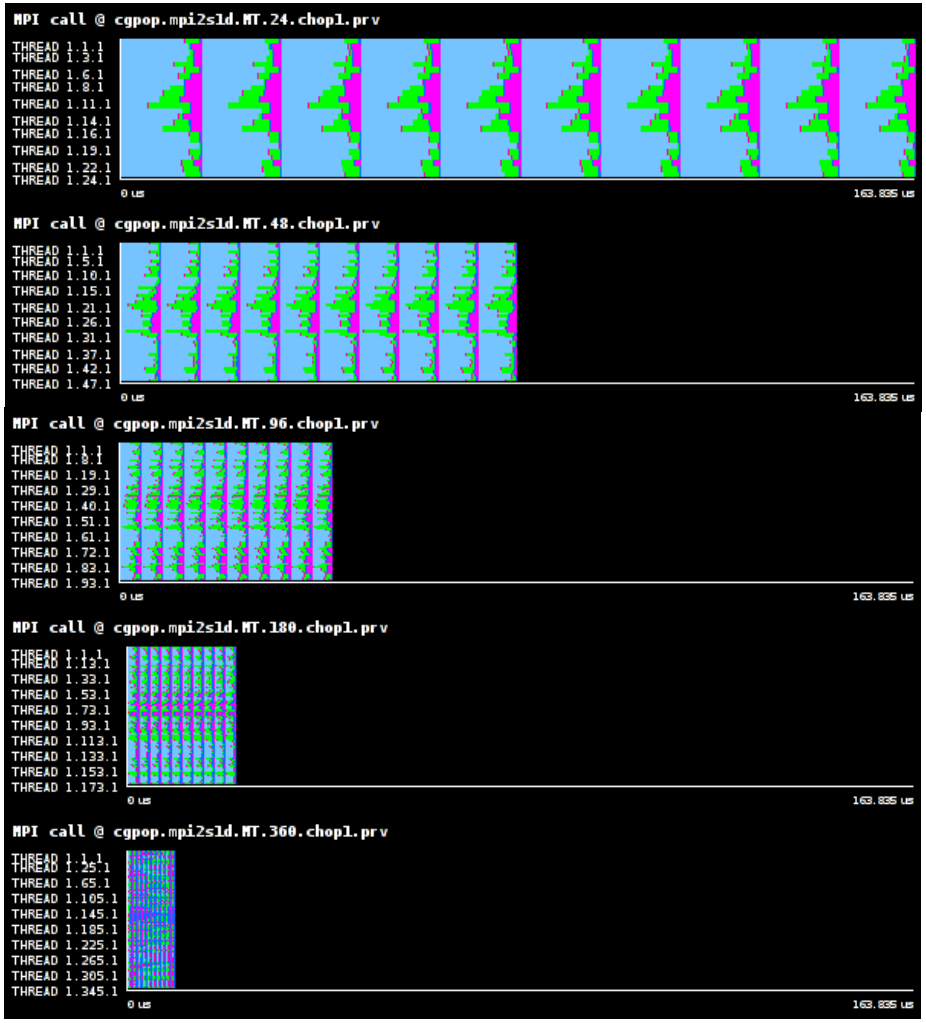
24 tasks

48 tasks

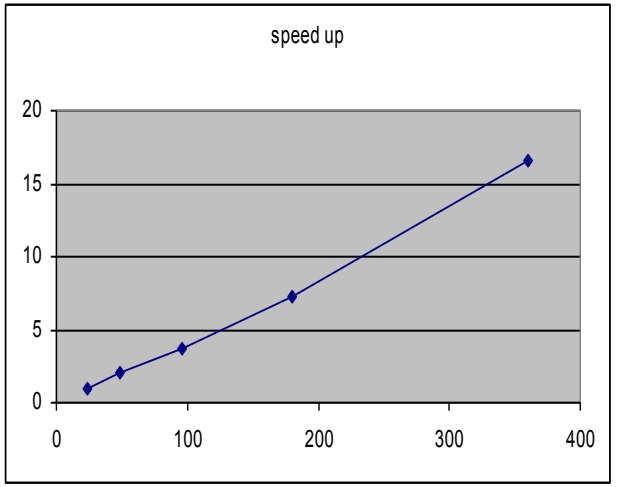
96 tasks

180 tasks

360 tasks

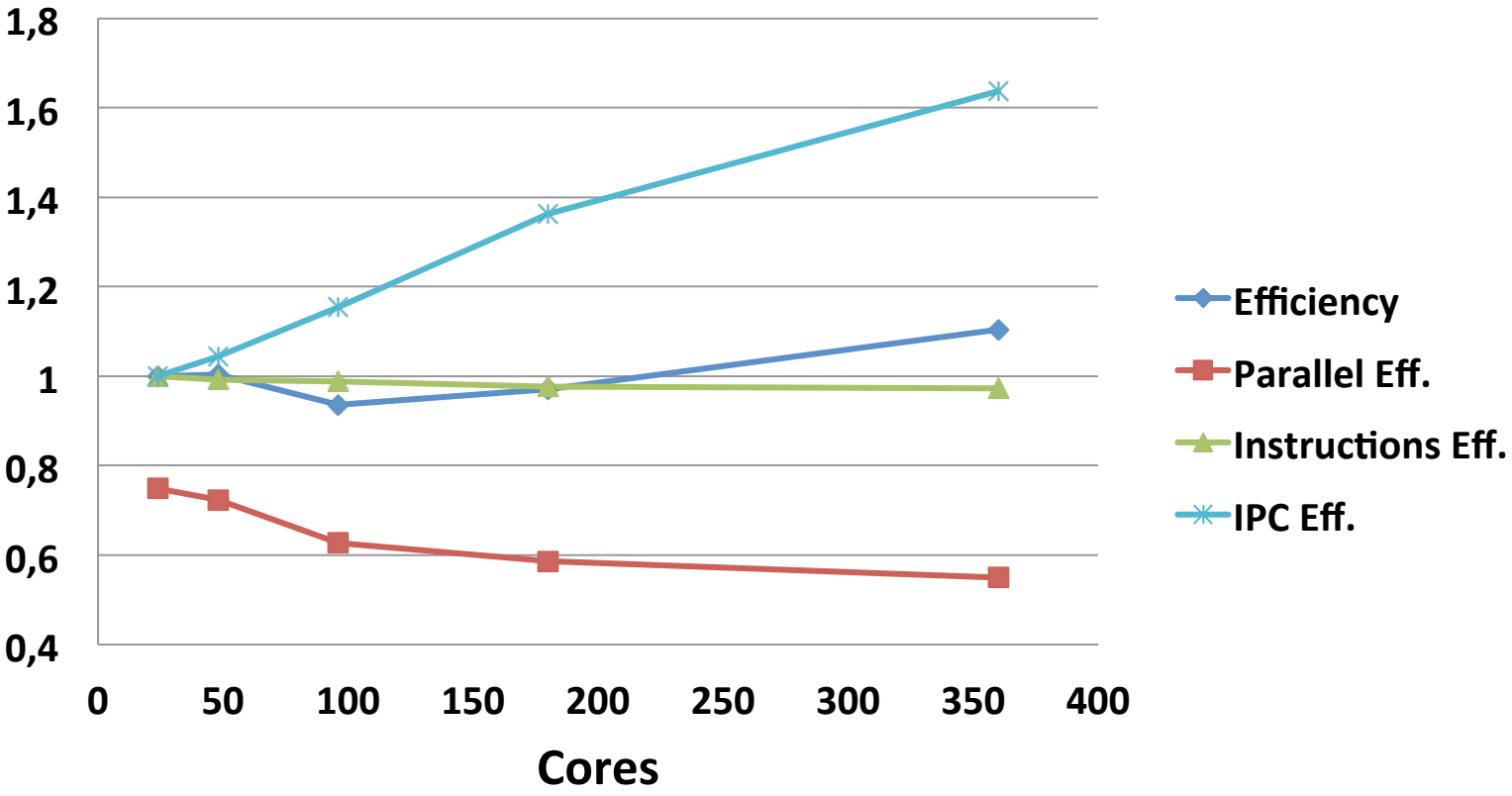


Good scalability!



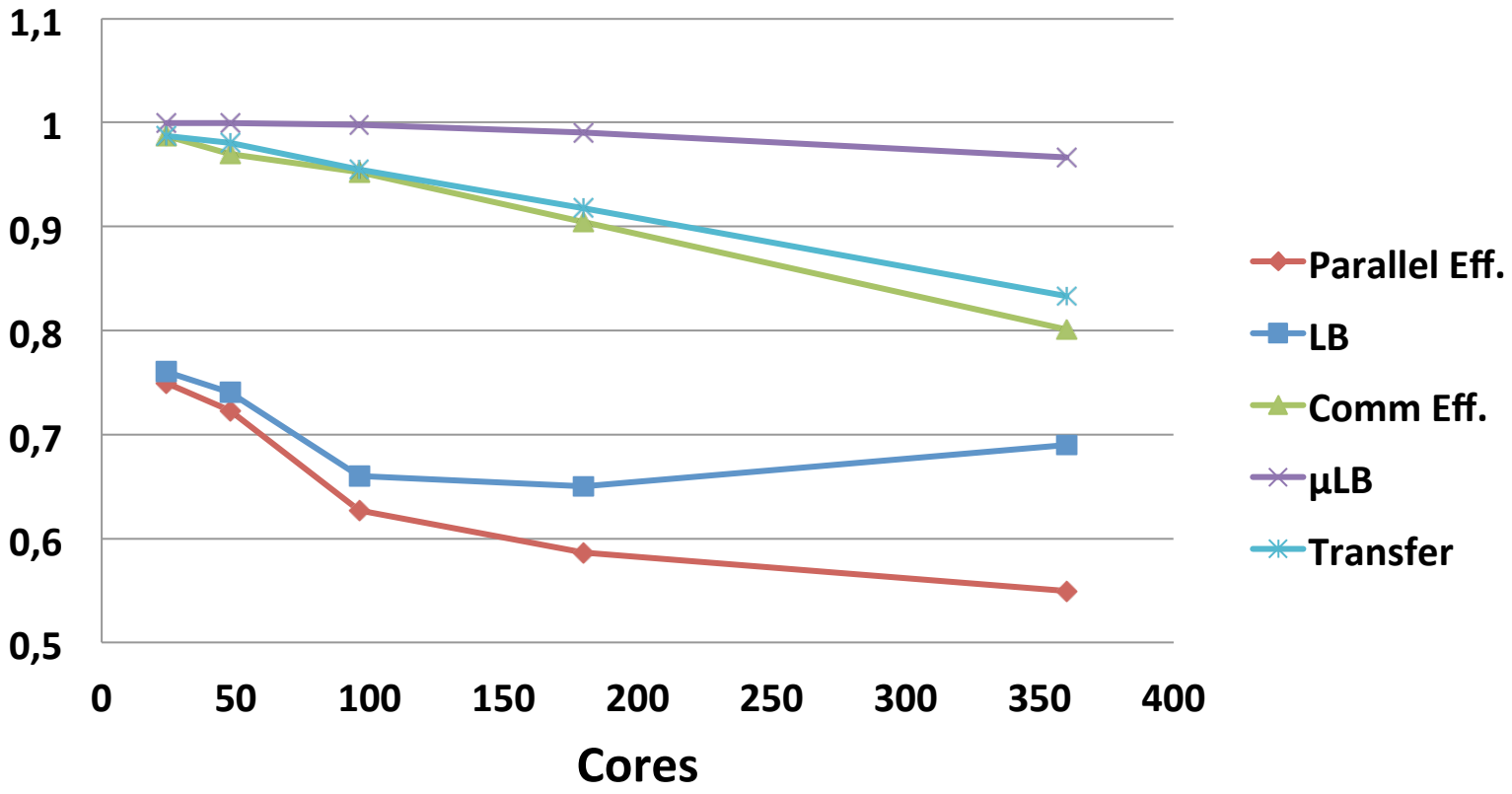
Scaling model results (I)

Comparing 10 iterations program global efficiency



Scaling model results (II)

Comparing 10 iterations parallel efficiency

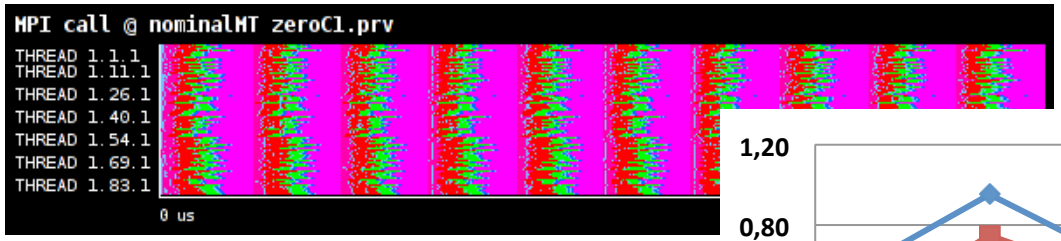


Imbalance

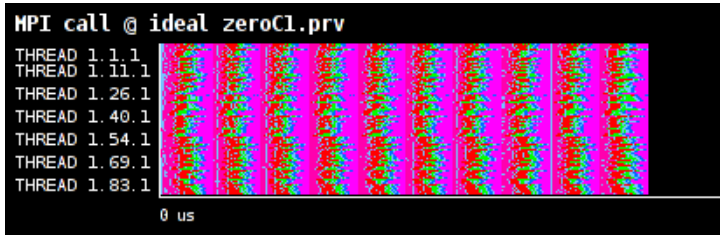
Communication phase only (96 Tasks)

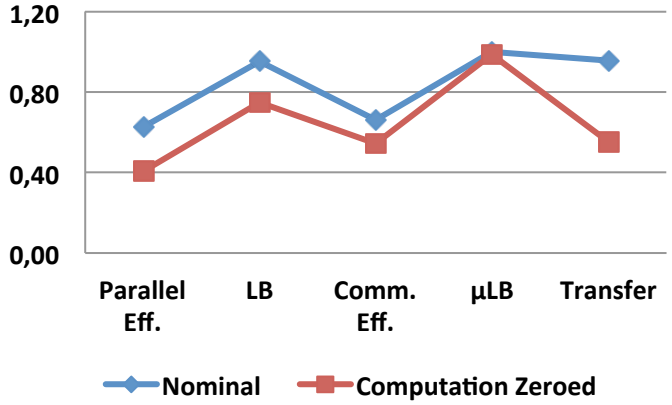
- Small computations point-to-point / collective calls
- MPI calls view / same time scale

Nominal
Main computation = 0

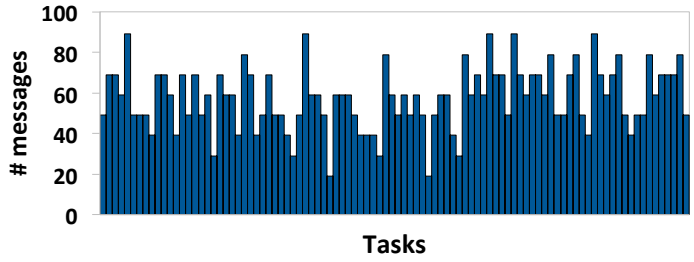


Ideal network
Main Computation = 0





- Imbalance in halo exchange
  - Computation (75%) / # messages (65%)
- Small amount of serialization
- Large transfer time



CONCLUSIONS

What did we learned?

- How network factors affect my application?
- Does it make sense to use asynchronous communications?
- Which is the best combination of computing power and network speed?
- Does my application take into advantage the resources available?
- Is the parallelization of my application efficient?

Wrapping up

- Sometimes things are not as they look like
- Small scale analysis can give us hints to identify problems at larger scales
- DIMEMAS offers a valuable insight to understand our applications and systems

www.bsc.es

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

THANK YOU!

tools@bsc.es

PATC Tools Training. Barcelona,
Oct. 14-18th 2013