Research Activities of the GPPD

Parallel and Distributed Processing Group

Load Balancing Strategies

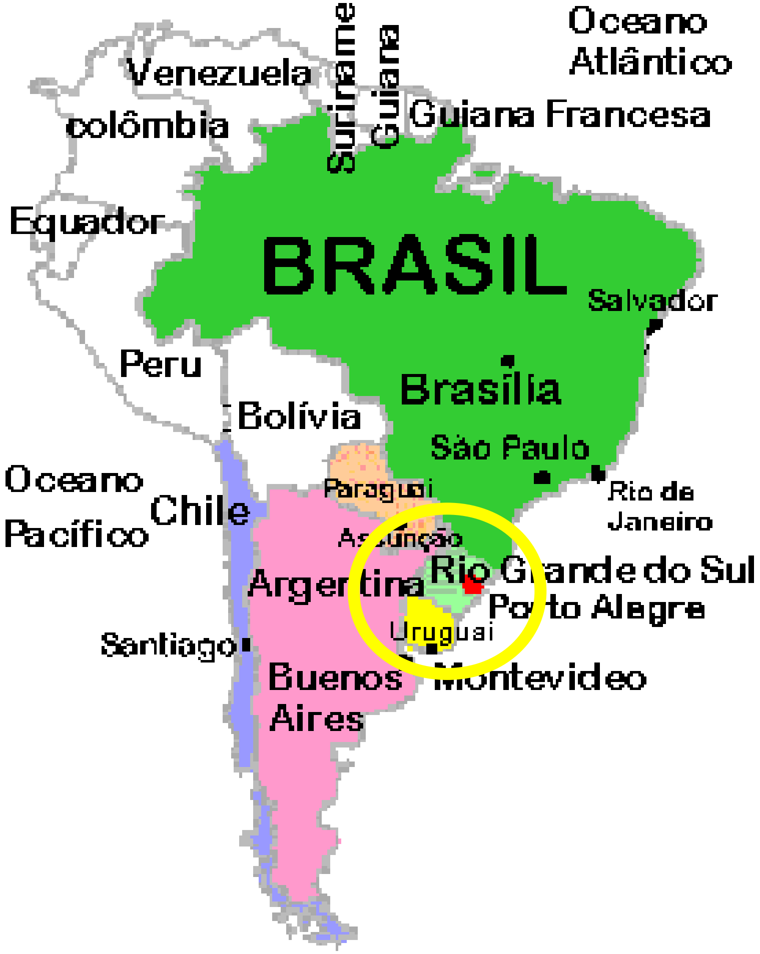
Informatics Institute
UFRGS

Philippe O. A. Navaux

Third Workshop of the CNPq-Inria HOSCAR Project

Bordeaux, September 03, 2013

We are located in Brazil

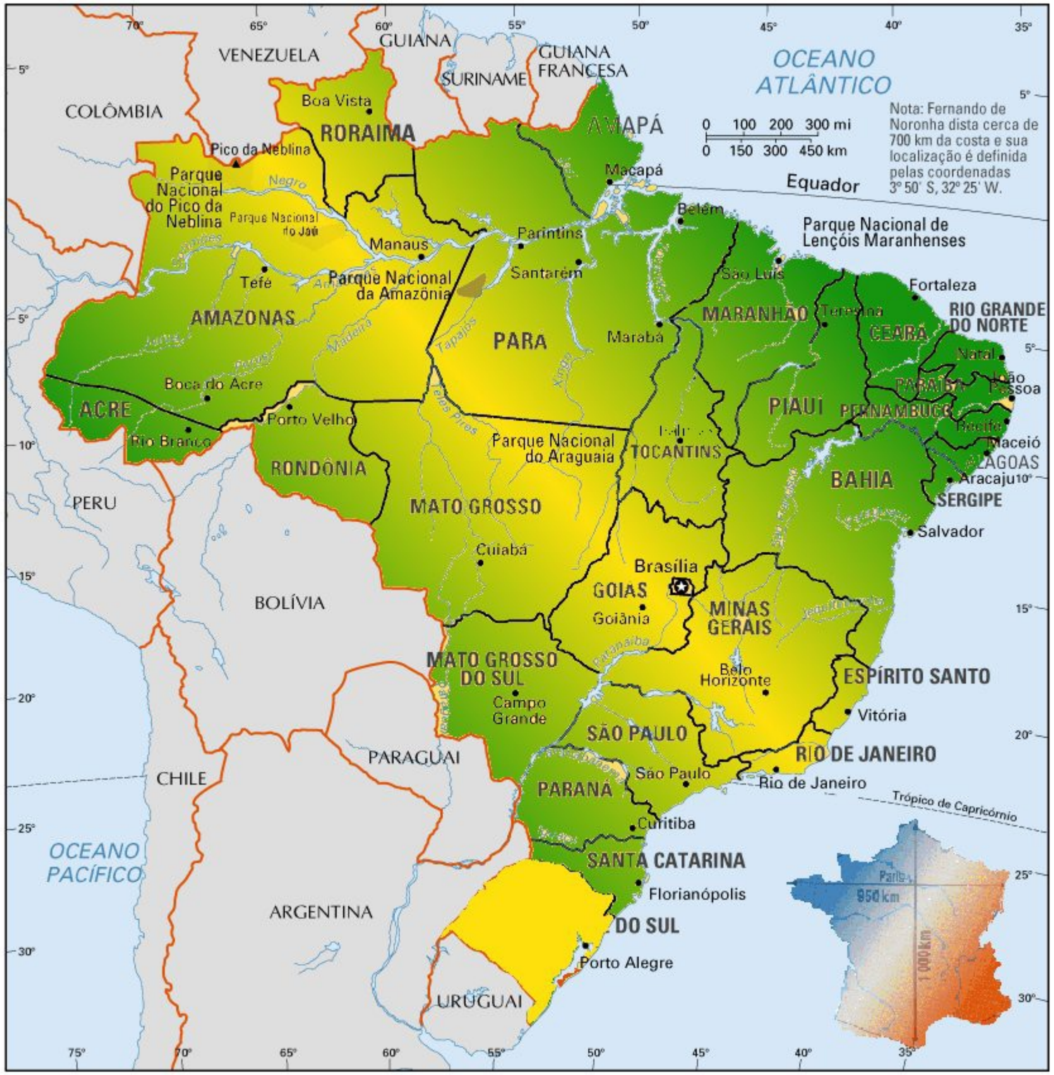


… in the State of
Rio Grande do Sul

.inf INSTITUTO DE INFORMÁTICA UFRGS

Porto Alegre, RS, Brasil

UFRGS UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL



GPPD
Grupo de Processamento Paralelo e Distribuído

Porto Alegre

Population: 1.44 Million

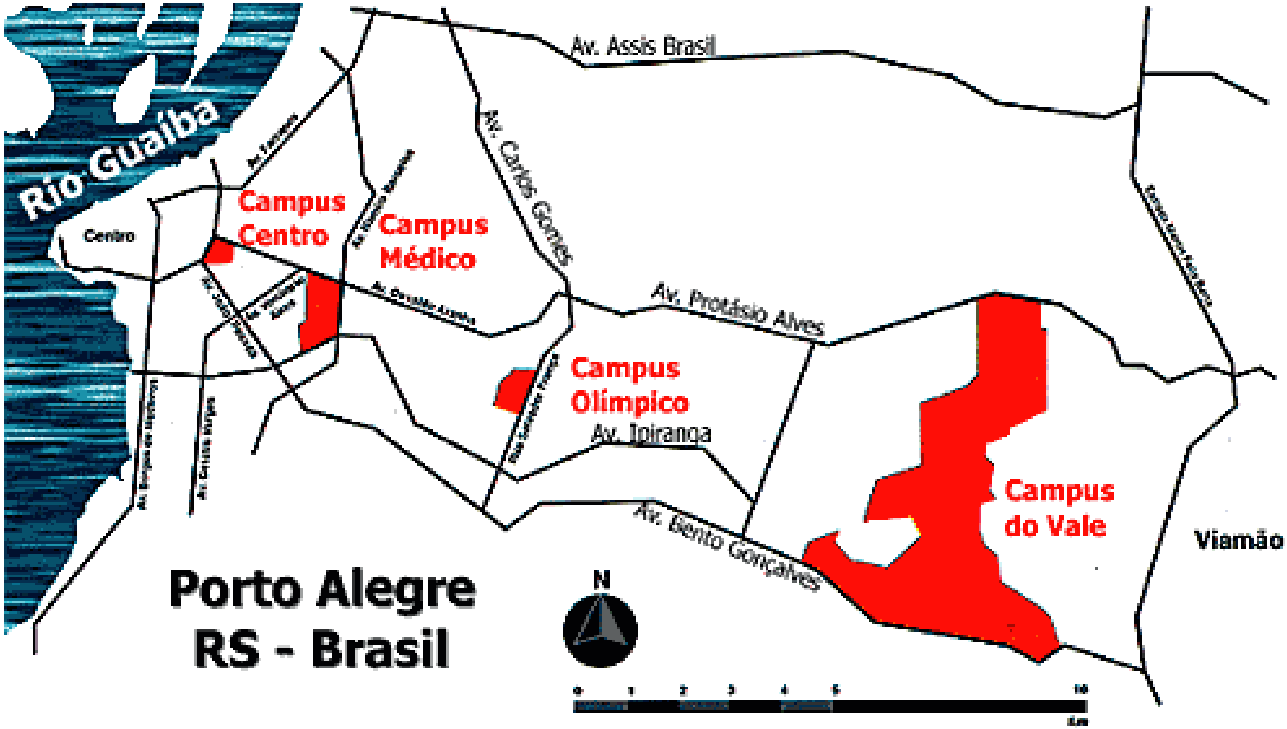
Climate: S

four well

Gr

UFRGS
Federal University of Rio Grande do Sul

- Created in 1.895
- One of the five principal universities in Brazil
- Approximately 2.278 faculty members
- Students: approximately 30.000 (undergraduate and graduate)
- Four campi



The Institute in Numbers

74 Full-time Professors

2 Departments

- Theoretical and Applied Informatics

2 Undergraduate Programs

- Computer Science and Computer Engineering

3 Graduate Programs

Masters and PhD in Computer Science, Microelectronics, Informatics in Education

691 Undergraduate Students

250 Graduate Students

The Institute in Numbers (Cont.)

4 Buildings

Total Area: 7,600 m^2

37 Laboratories

Including Teaching and Research Labs

1 Center of Events

4 Auditoria

1 Library

Area: 633 m^2

1 Company Incubator (CEI)

Faculty

- 74 full-time professors dedicated to research and education

- The largest group of researchers in Computer Science and Engineering in Brazil

Faculty members graduated from important institutions around the world

Brazil (26)
France (16)
German (11)
United Kingdom (5)
USA (3)

Belgium (2)
Portugal (2)
Canada
Switzerland

.inf INSTITUTO DE INFORMÁTICA UFRGS

Research

UFRGS UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

- Covers all traditional areas of Computer Science and Engineering, from Microelectronics to Computing Theory

Strong integration among hardware and software groups

Artificial Intelligence
Bioinformatics
Computer Graphics
Computer Networks
Databases
Distance Learning
Embedded systems
Fault tolerance
Formal Methods
Information Systems
Microelectronics
Multimedia
Parallel Processing
Robotics
Software Engineering

Technology Generation

- Continuous interaction with IT companies in the State
 - Origin of many software and most important hardware companies of the region
 - Origin of CEITEC the Microelectronic center of Rio Grande do Sul

Digitel
Altus
Digistar
Teracom

CP Eletrônica
Perto
Parks
and many others

Presenting the GPPD - Parallel and Distributed Processing Group

Main Research Areas

- Processors architectures
  - Thread Mapping, cache
- Power Consumption
- Cluster, Grid and Cloud computing
  - Virtualization
- Mobile and Ubiquitous Computing
- Parallel and distributed algorithms
- Performance evaluation
- Tools and environments for parallel and distributed programming
- Debugging and Monitoring

People GPPD

2006

Professors
Alexandre Carissimi
Cláudio Fernando Resin Geyer
Nicolas Maillard
Philippe Olivier Alexandre Navaux
Tiarajú Asmuz Diverio

PhD Students
Cristiano André da Costa
Débora Nice Ferrari Barbosa
Eduardo Rocha Rodrigues
Emerson André Fedechen
Fábio Reis Cecin
Henrique Cota de Freitas
Lucas Mello Schnorr
Luciano Cavalheiro da Silva
Marcos Ennes Barreto
Marko Petek
Mozart Lemos de Siqueira
Márcia Cristina Cera
Mônica Xavier Py
Rodrigo da Rosa Righi

Master Students
Alexandre da Silveira Ilha
Caciano dos Santos Machado
Carlos Eduardo Benevides Bezerra
Clarissa Cassales Marquezan
Eder Stone Fontoura
Elton Nicoletti Mathias
Everton Hermann
Guilherme Peretti Pezzi
Gustavo Romano
Gustavo Cestari Frainer
João Vicente Lima
Luiz Sequeira Laurino
Marcelo Veiga Neves
Marco Antonio Zanata Alves
Rafael Keller Tesser
Rodrigo Virote Kassick
Rômulo Bandeira Rosinha
Sonia Andrea Lugo Vazquez

Associated Researchers
Patrícia Kayser Vargas Mangan
Rafael Bohrer Ávila
Roberto Pinto Souto
Tatiana Gadelha Serra dos Santos

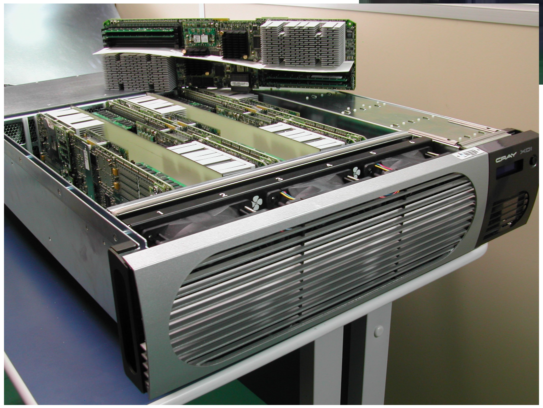
Undergraduate Students
Danilo Fukuda Conrad
Eduardo Dias Camaratta
Francieli Zanon Boito
Laércio Lima Pilla
Manuela Klanovicz Ferreira
Vicente Silva Cruz

People 2011

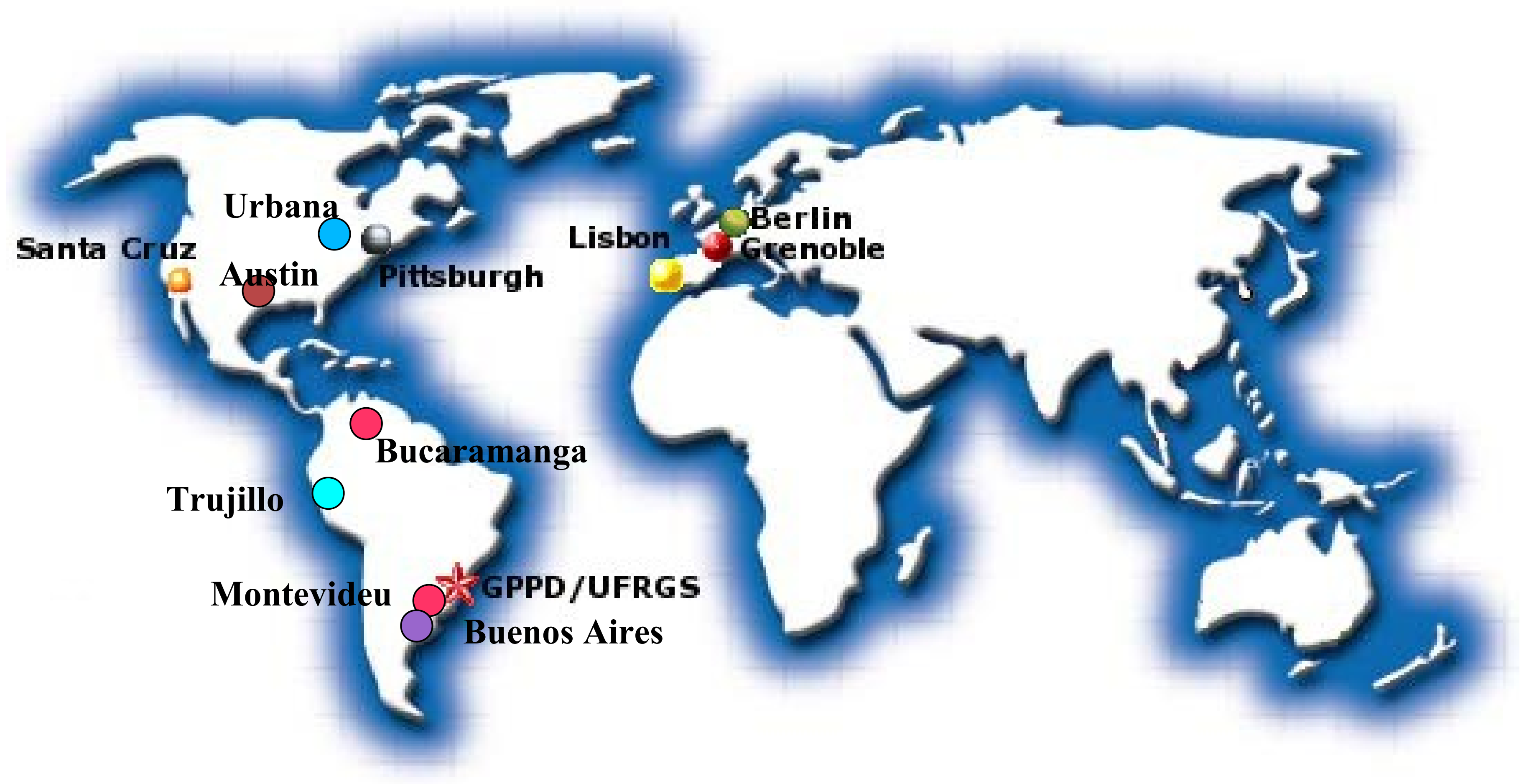
The group (12/2012)

Clusters

- Clusters:
  - DELL Cluster 112 cores,
    - 14 nodes with 2 quad Xeon processors
  - CRAY XD1
    - 6 nodes, dual Opterons with FPGAs.
  - DELL Cluster - LABTEC
    - 20 nodes, PIII, fast-ethernet
  - Itautec Cluster
    - 16 nós, Myrinet
- Numa Machines
- Access to Grid 5000
- Access to Cray Machines

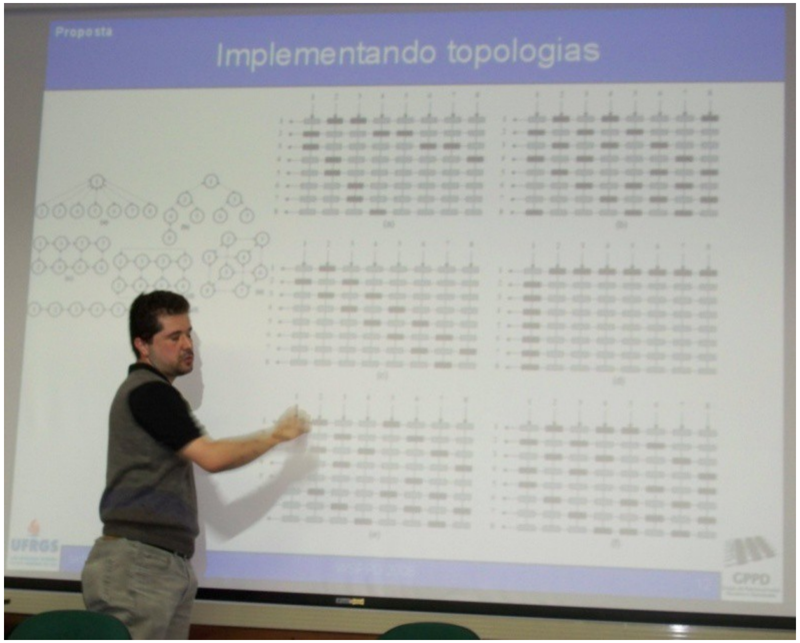


International Cooperation



Some research in GPPD

Architecture



Henrique Cota

Marco Zanatta

Enhancing Energy Efficiency using Efficient Parallel Programming Techniques
Thank you
Suported by

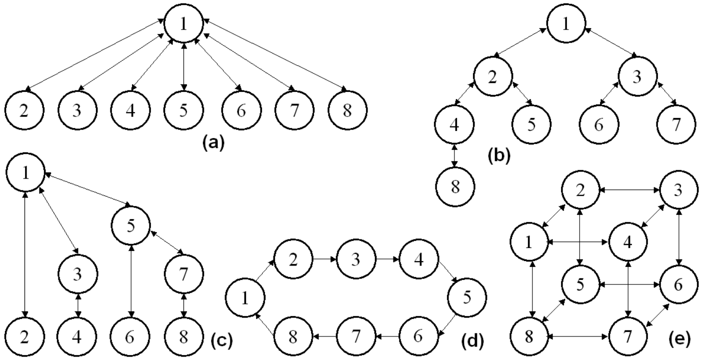
Network on chip,
Memory Hierarchy,
Energy efficient parallel hardware…

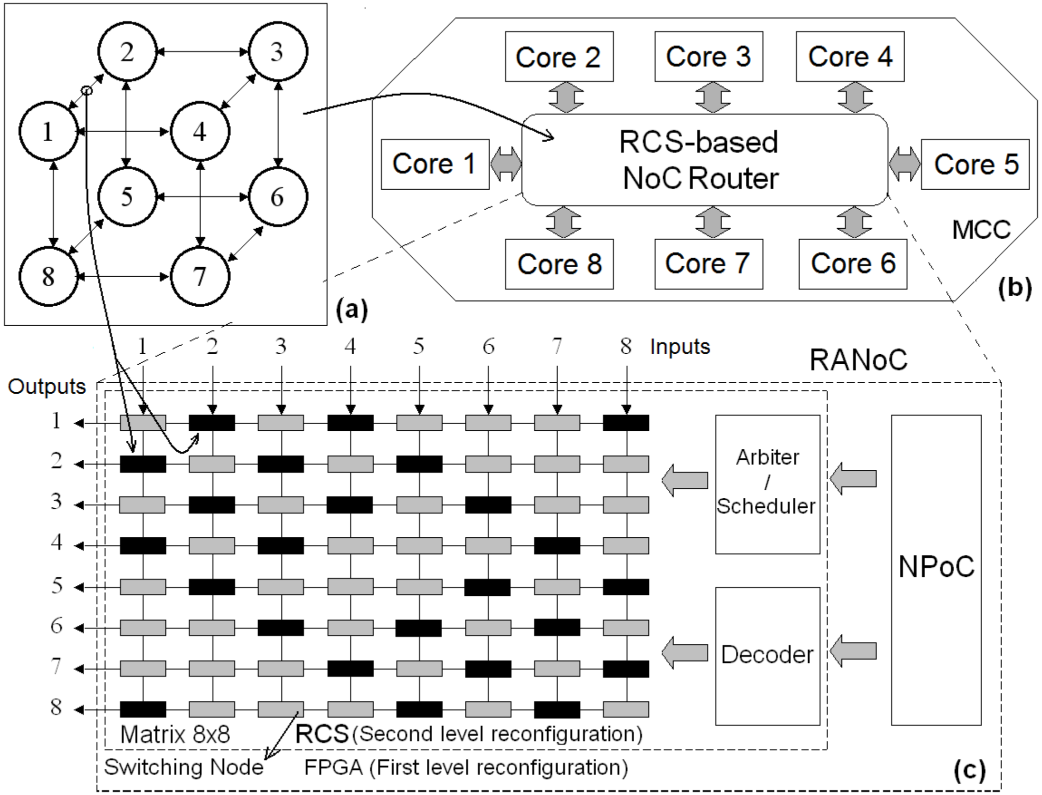
Architecture: Network on Chip

- NoC

  Network on Chip

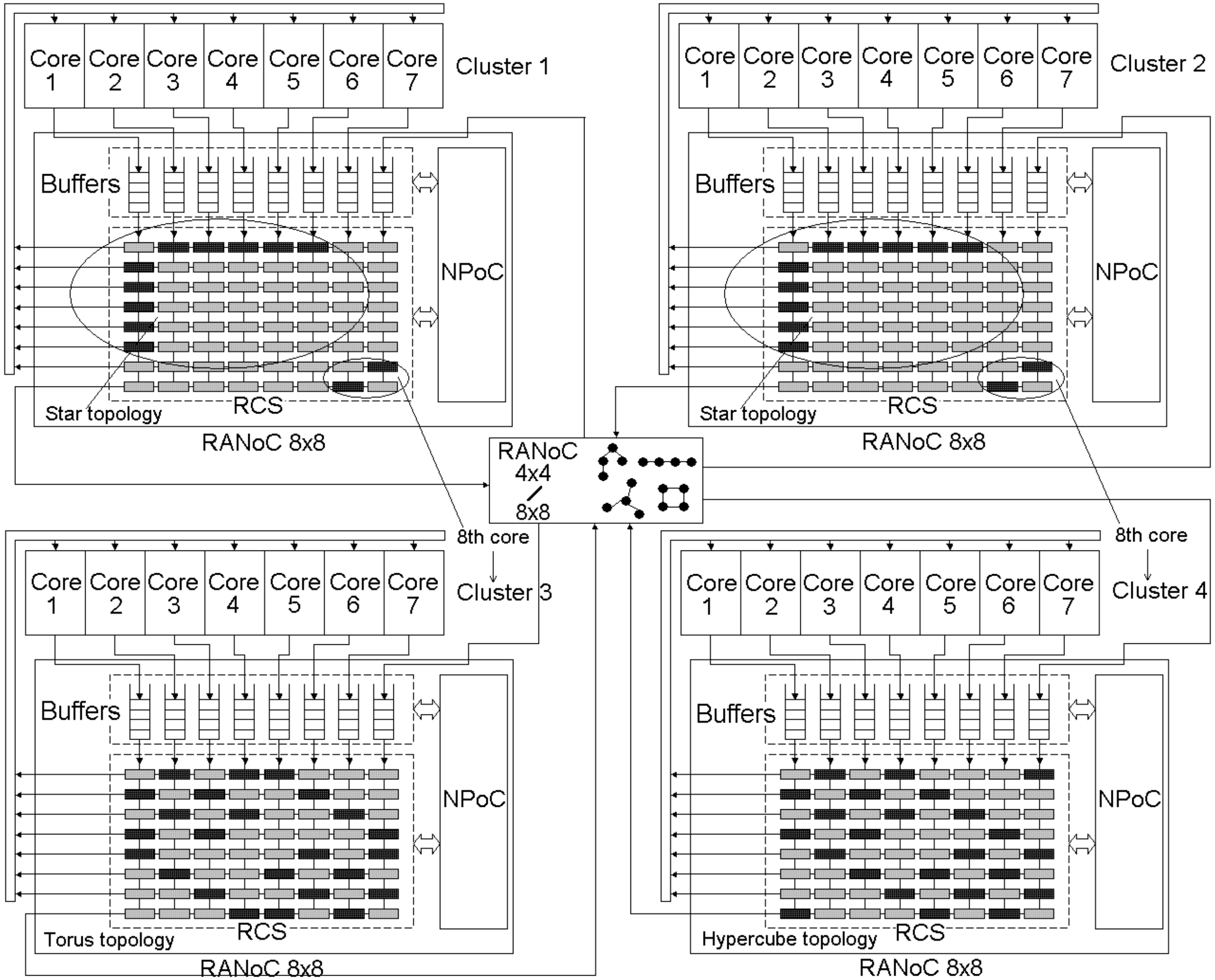
  – Henrique Costa Freitas





Architecture: Cluster on a chip

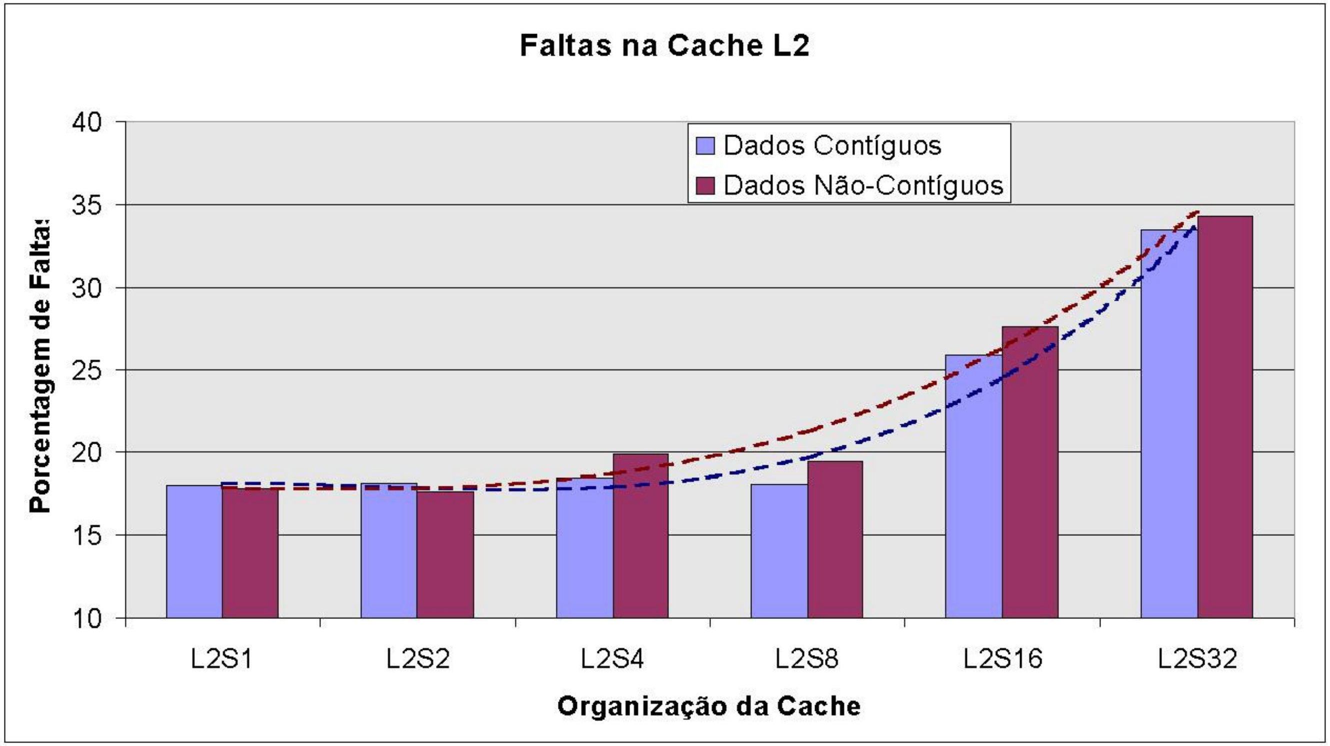
- Multi Core Clusters on a Chip
  - Henrique Cota Freitas



Architecture: Influence of Cache L2

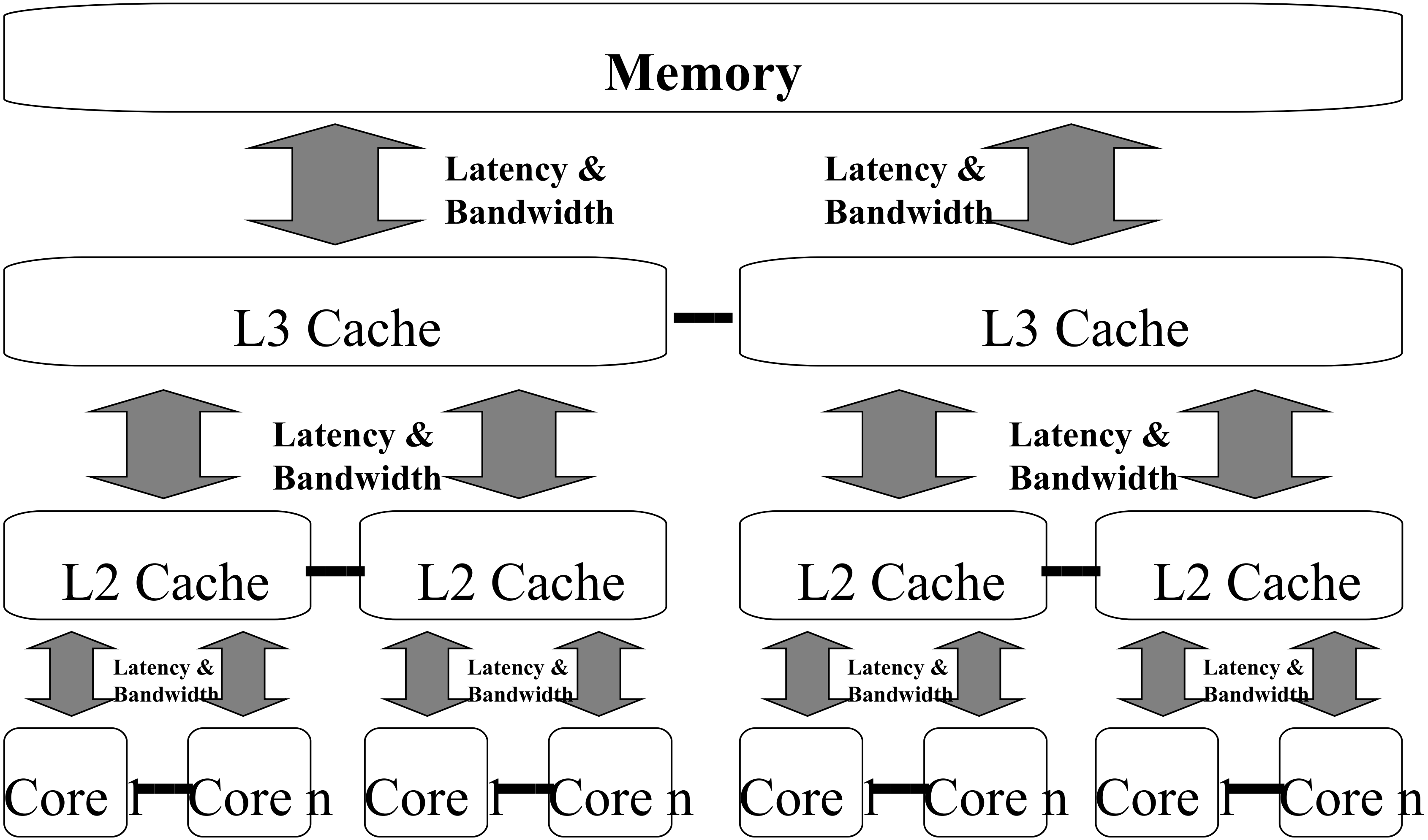
- The influence of sharing L2 Cache in a Multiprocessor Chip
  - Marco Zanata Alves
  - Models for L2Cache:
    - Completely shared ?
    - Shared by groups/ clusters ?

- Actually using L2
  - Niagara – SUN
  - Power5 – IBM
  - Intel, AMD, etc



Architecture:
Levels of communication on Multi-core

Manuela Ferreira



Power Consumption

- Questions:
  - Which is impact on power consumption of a virtual machine instance?
  - How frequency reduction influences application performance and power consumption on virtualized system?

NUMA – thread and data mapping

- NUMA challenges
  - Memory accesses to remote nodes
  - Contention on interconnections
  - Communication across memories

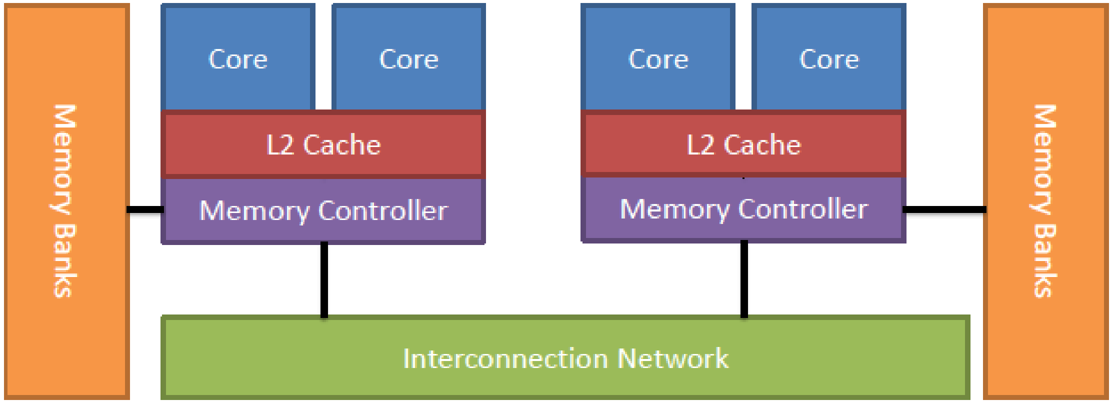
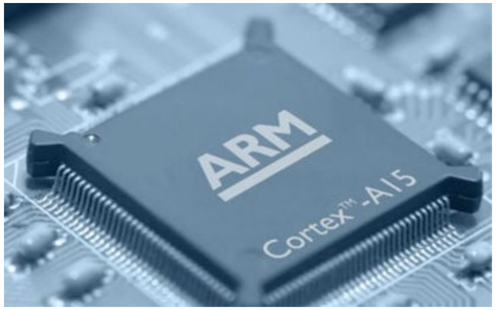


Fig.1 NUMA Architecture

- Data distribution is important
- Solution: manage memory and thread affinity
- Goal: optimize memory access latency and bandwidth for the whole application

Low-power processors for HPC



- Energy challenges:
  - Minimize energy consumption.
  - Attain performance restraints.
- Solution: Include low power processors as an HPC resource.
- Goal: Efficient utilize low power and high performance processors to improve the energy efficiency.

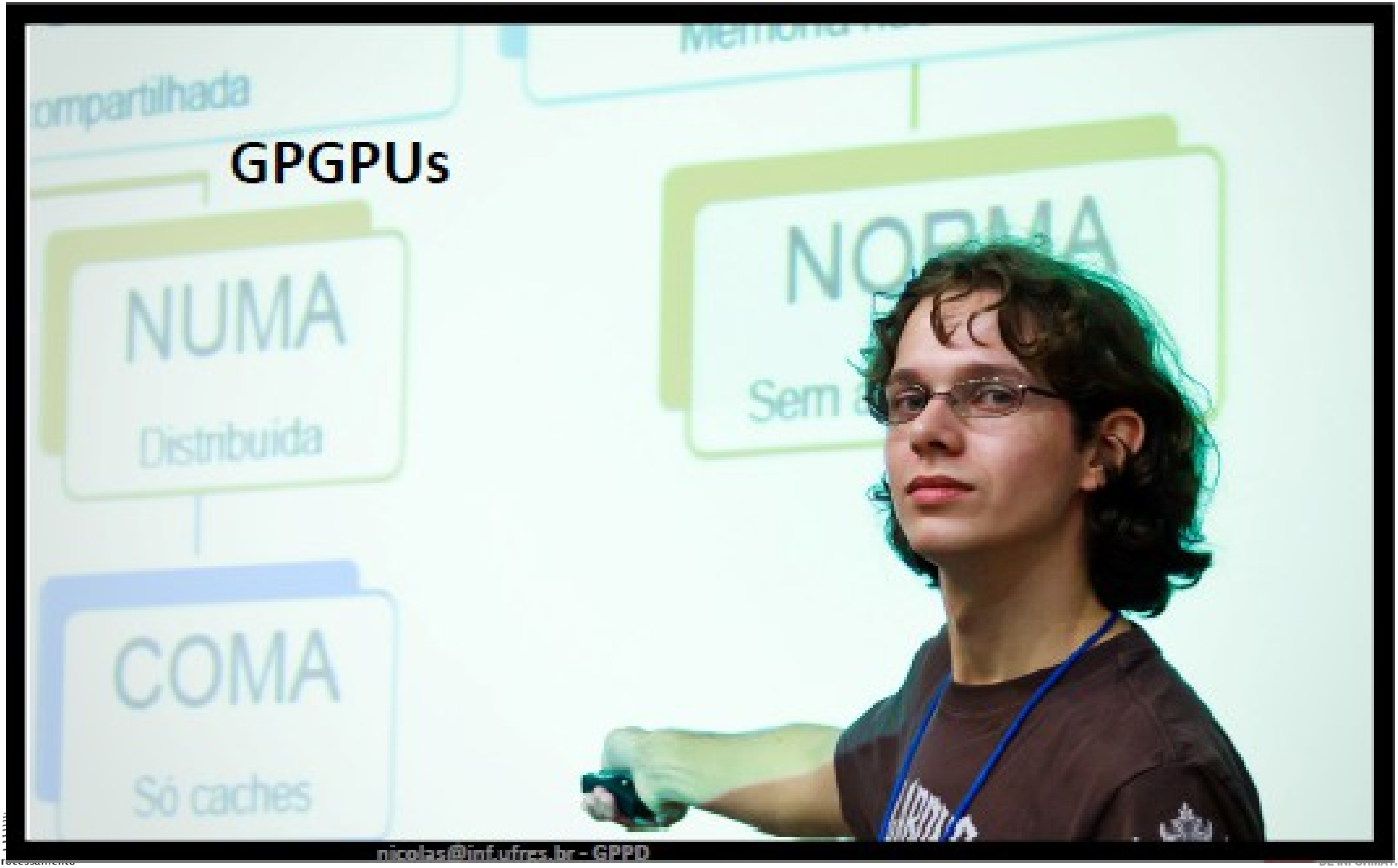
Cluster of ARM Processor

Low-power processors for HPC

- Question:
  - How interesting is to employ ARM processors on HPC application? Which are the main limitations?
- Experimental platform:
  - 8 nodes clusters (pandaboards)   ARM 9 dual-core (1.2 GHz), 1 GB RAM (DDR 2)   Linux Kernel
- People:
  - Rodrigo K. Ferreira (undergrad student,
- Advisor: Alexandre Carissimi
- Evaluate network bandwidth
  - Netperf benchmark
- Evaluate MPI applications and their limitation on ARM processor clusters (memory)
  - NAS Benchmarks (MPI)

Heterogeneous HPC

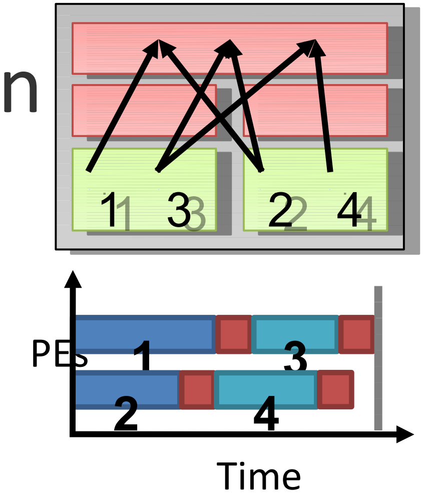
Laércio Pilla

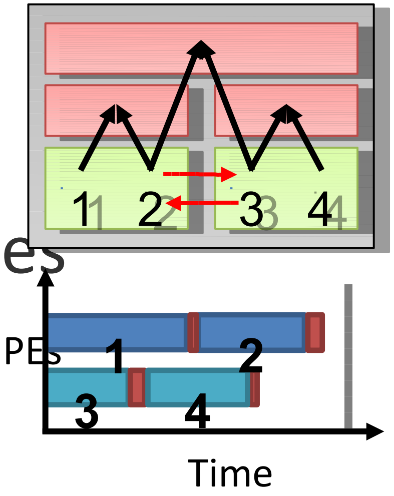


nicolas@inf.ufrgs.br - GPPD

Heterogeneous: Runtimes for new HPC Platforms

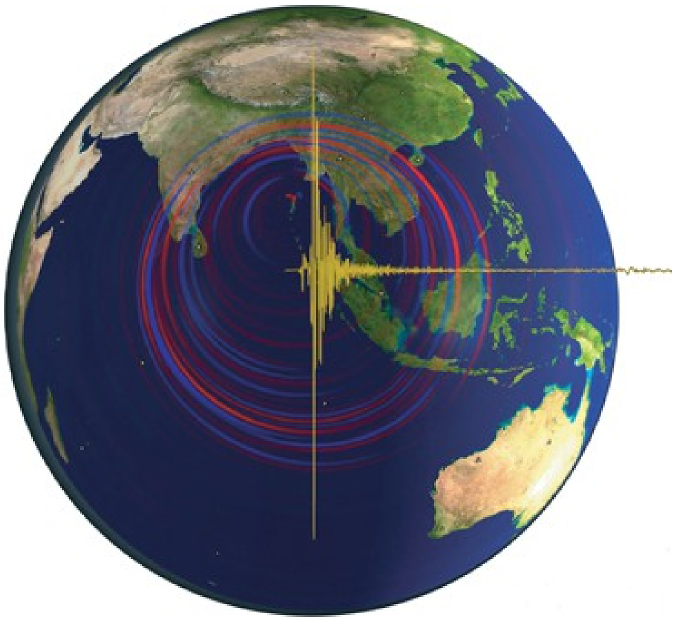
- Scheduling and data distribution
  - NUMA machines
    - NUMA factor
    - [Cache] memory latencies
    - Cache misses
  - Clusters of NUMA machines
  - Heterogeneous/hybrid machines
    - CPU + GPU
  - ARM





Heterogeneous: Runtimes for new HPC Platforms

- Case studies
  - SPECFEM3D
  - Ondes3D
- Tools
  - Charm++
  - Hwloc
  - SGPU_2
  - StarPU

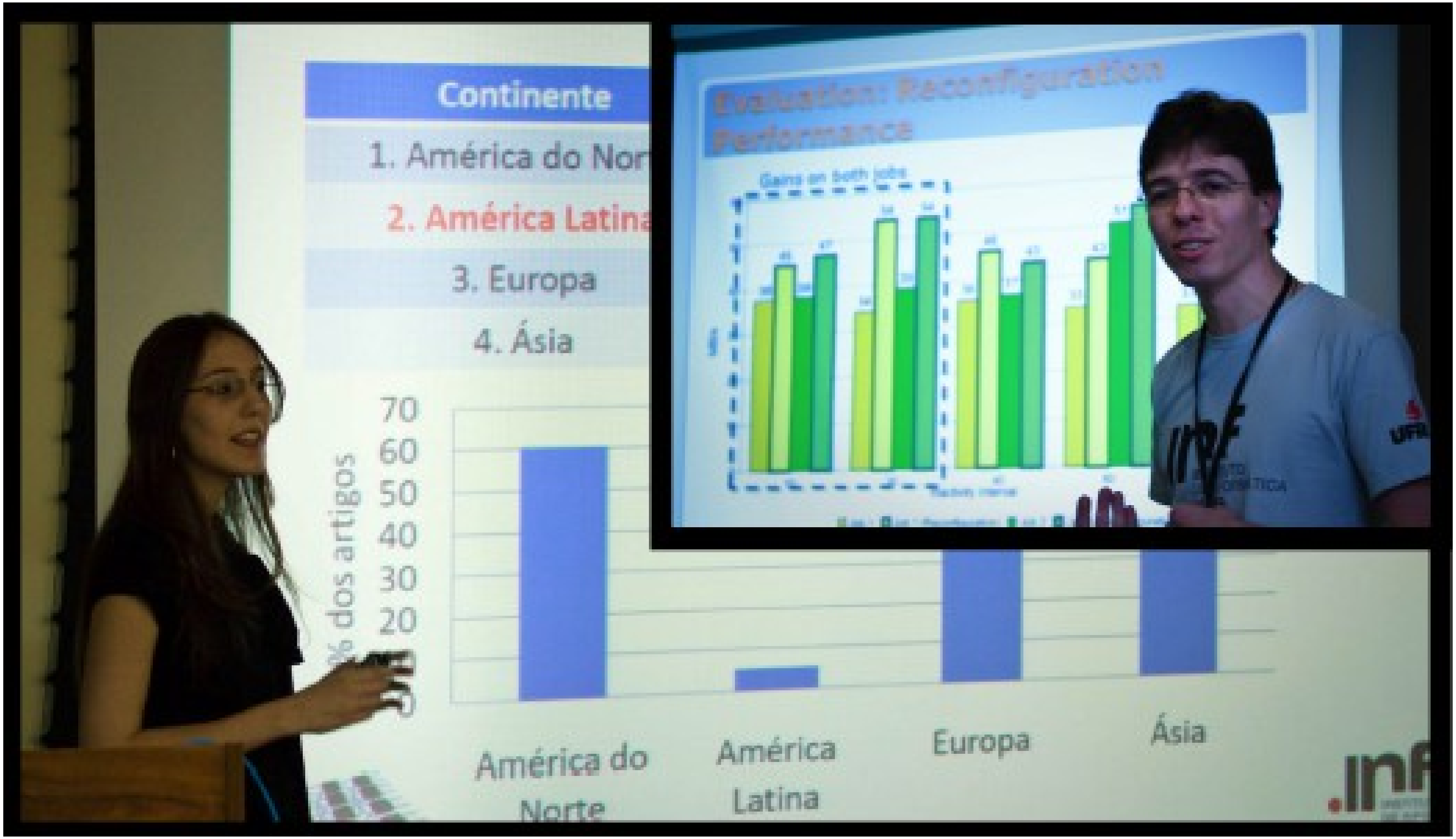


Dimitri Komatitsch, http://komatitsch.free.fr/

Parallel IO & Storage

Francieli Zanon

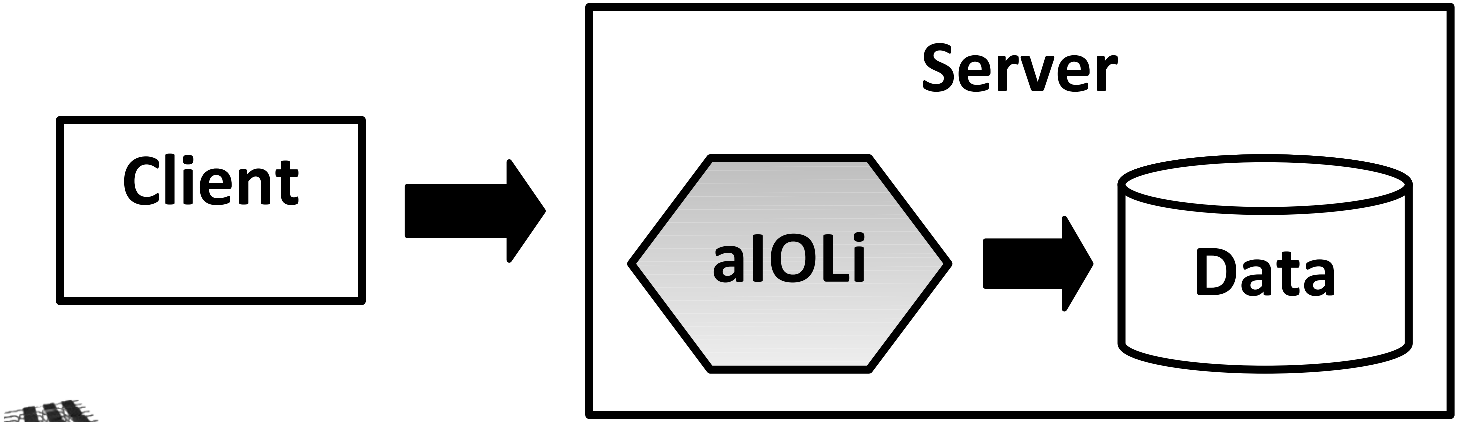
Rodrigo Kassick



I/O Scheduling

The goal: **To investigate the scheduling of I/O operations on parallel file systems**

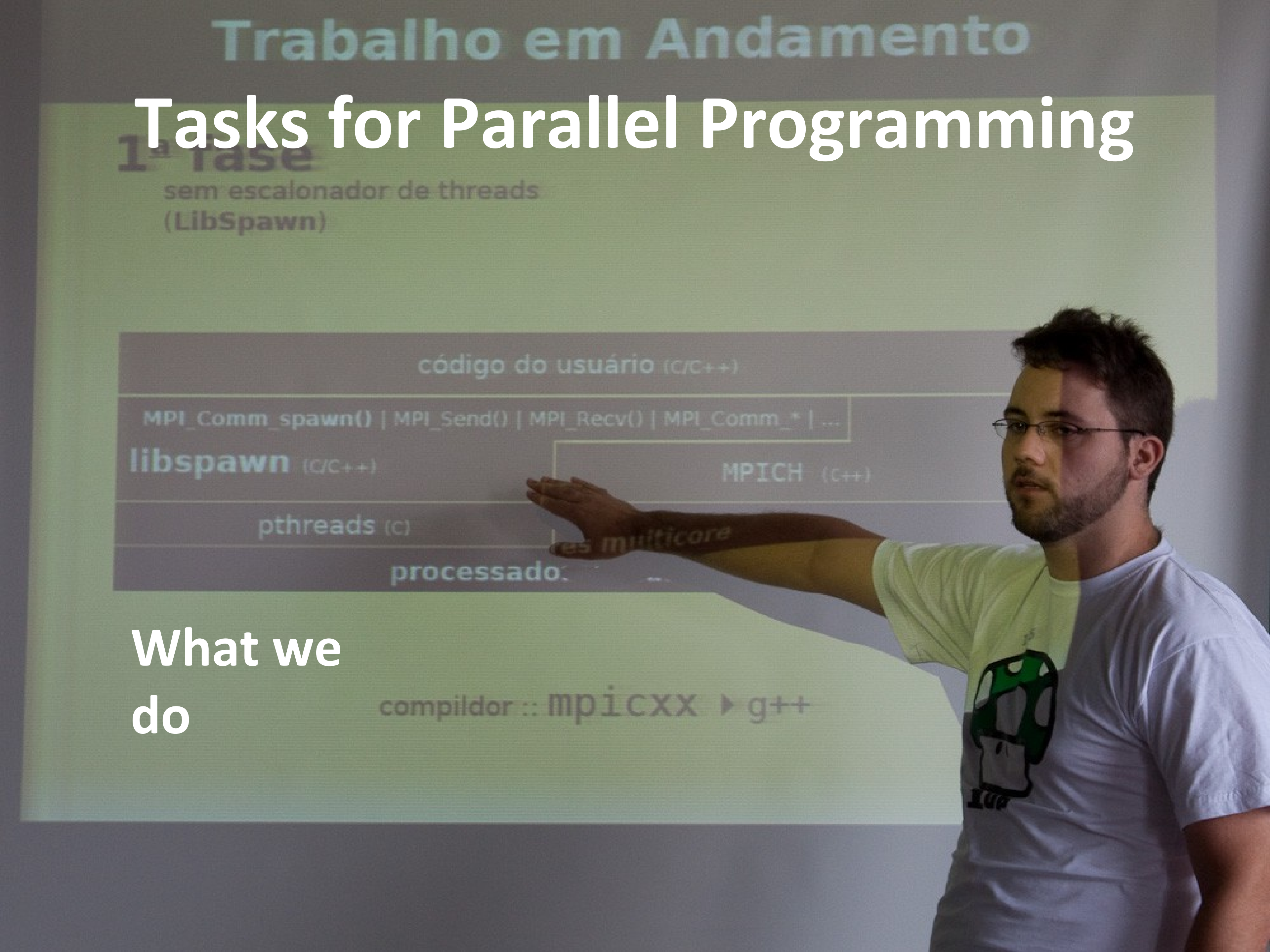
- The tool: aIOLi (Lebre et al., 2006)



I/O Scheduling

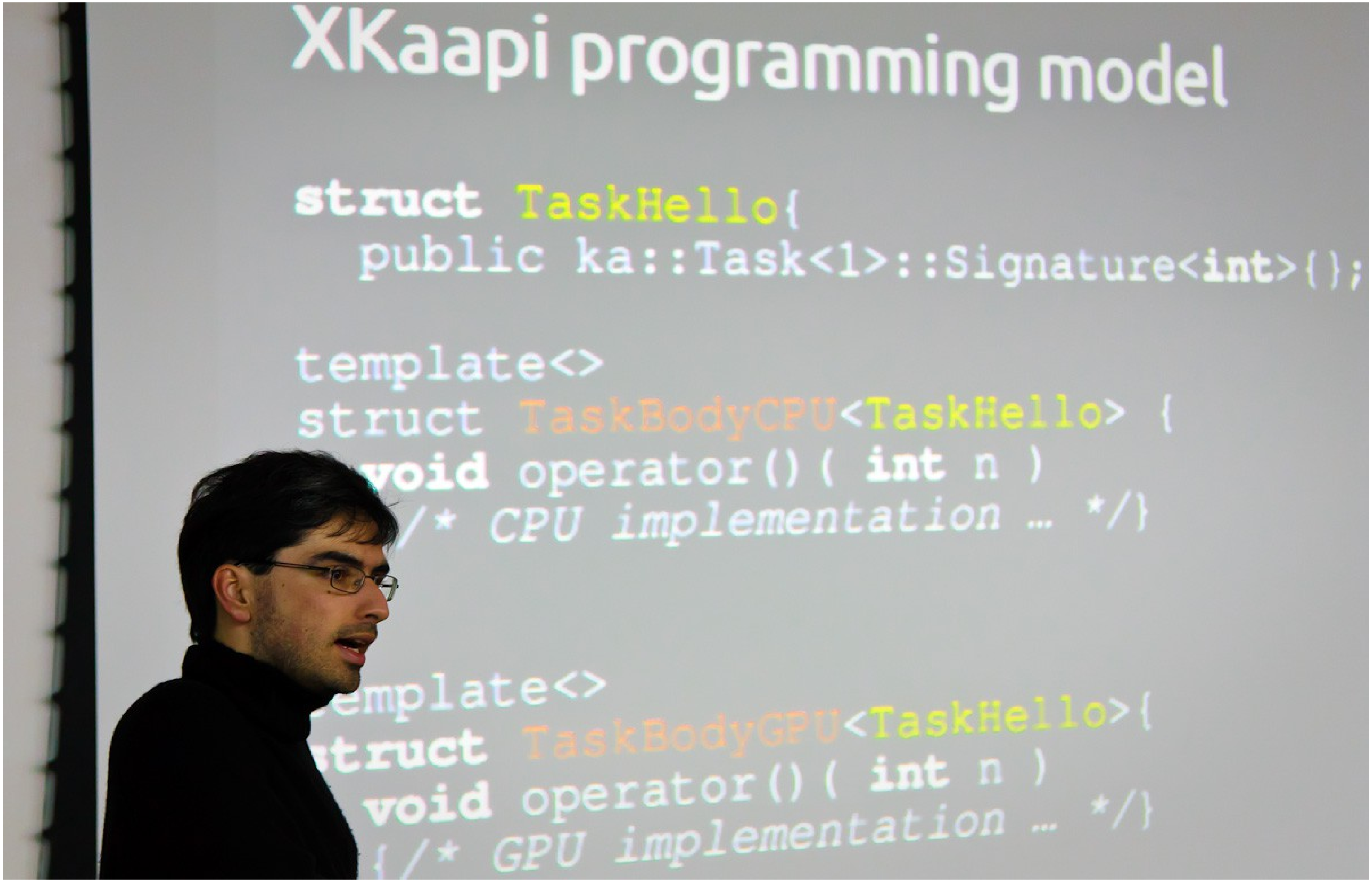
- How to improve the scheduler's decisions?
  - Hypothesis: **include information about the applications' access patterns**
- Subject of **Francieli Zanon Boito**'s PhD
  - Advised by **Philippe Navaux** and **Yves Denneulin**
  - 2010 to 2013

Tasks for Parallel Programming


Trabalho em Andamento
1ª fase
sem escalonador de threads
(LibSpawn)
código do usuário (C/C++)
MPI_Comm_spawn() | MPI_Send() | MPI_Recv() | MPI_Comm_* | ...
libspawn (C/C++)
MPICH (C++)
pthreads (C)
processado
es multicore
compildor :: mpicxx ▸ g++

What we do

Tasks with Kaapi



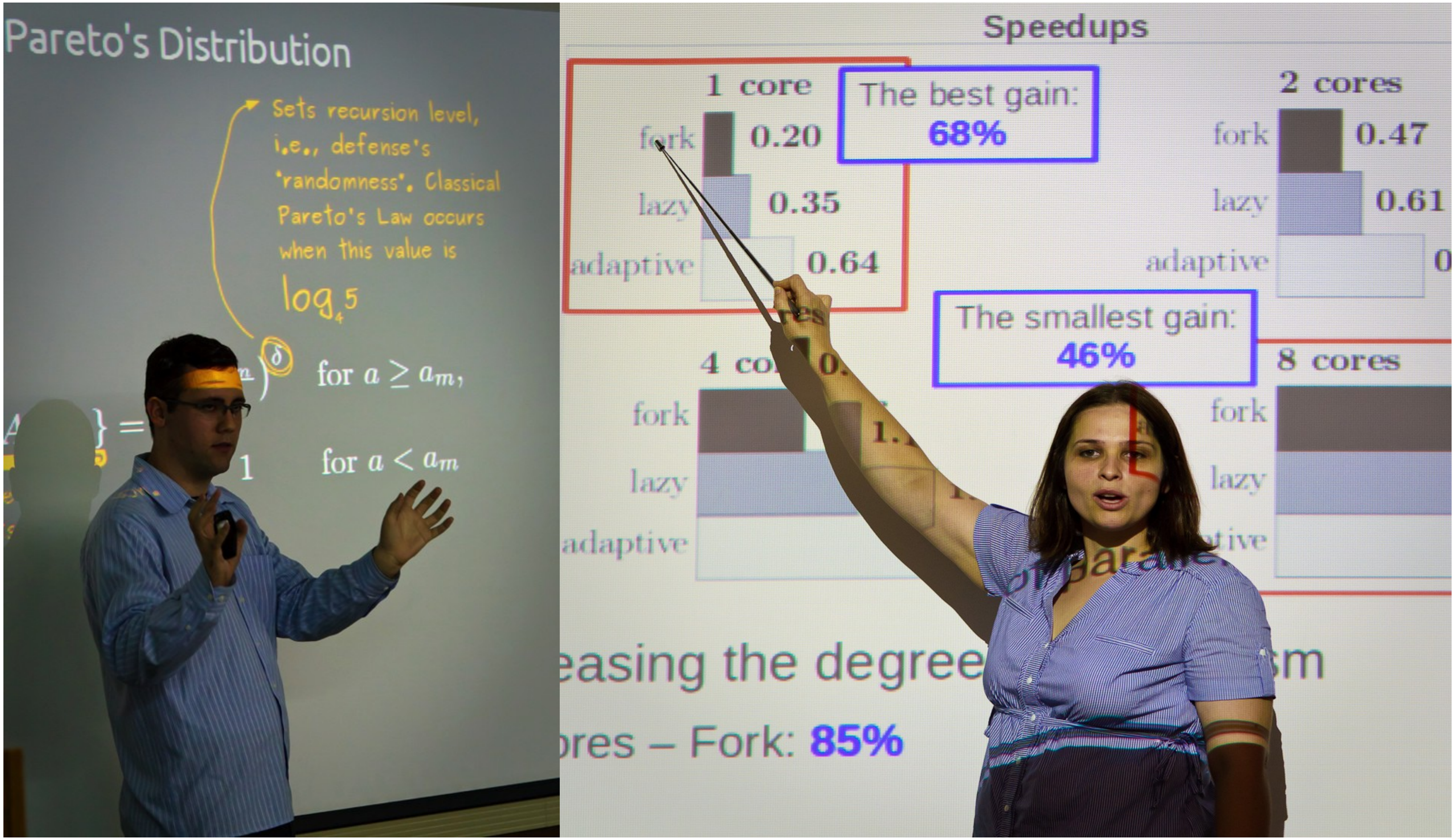
XKaapi programming model

```
struct TaskHello{
  public ka::Task<1>::Signature<int>{};

template<>
struct TaskBodyCPU<TaskHello> {
  void operator()( int n )
  /* CPU implementation … */}


emplate<>
truct TaskBodyGPU<TaskHello>{
  void operator()( int n )
  {/* GPU implementation … */}
```

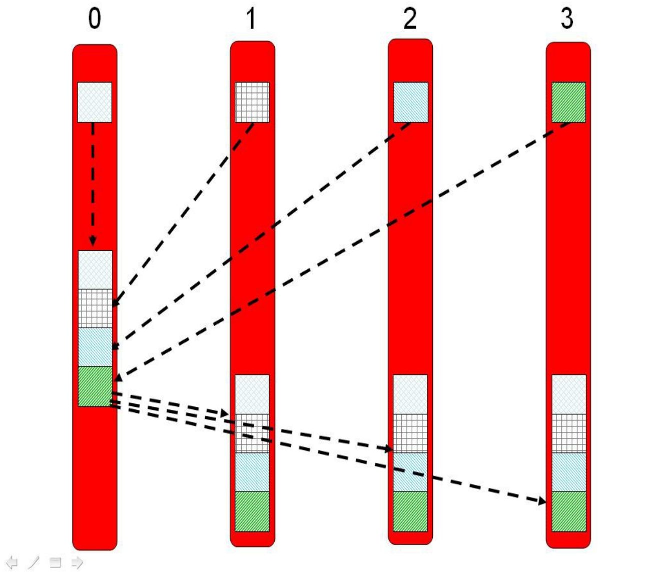
Kaapi

- A French project - http://kaapi.gforge.inria.fr/
- C++ library that provides an API for parallel programming based on tasks.
- A shared global address space
  - You create objects inside it with the keyword “shared”
- A task is a call to a function, prefixed by “fork”
  - Just like Unix / Cilk
  - Tasks communicate through objects *shared*.
  - Tasks specify the access mode to shared objects (read/write)
- The Kaapi runtime builds the data-flow graph and uses it to schedule the tasks.

Tasks with MPI?



The MPI task



- MPI defines tasks that:
  - Have their own address space,
  - Communicate with other tasks through messages.
  - Usually all are launched at the start of the program.
- The mapping "**MPI task**" **/ O.S.** is not specified.
  - Usually, 1 task == 1 heavy process (O.S. view);
    - MPI-2 has somewhat reinforced this common understanding
  - Some MPI Distributions use threads (MPICH);
    - A-MPI defines an abstract task (Urbana Champaign)

Dynamic Tasks in MPI: D&C and Spawn

- Using MPI-2

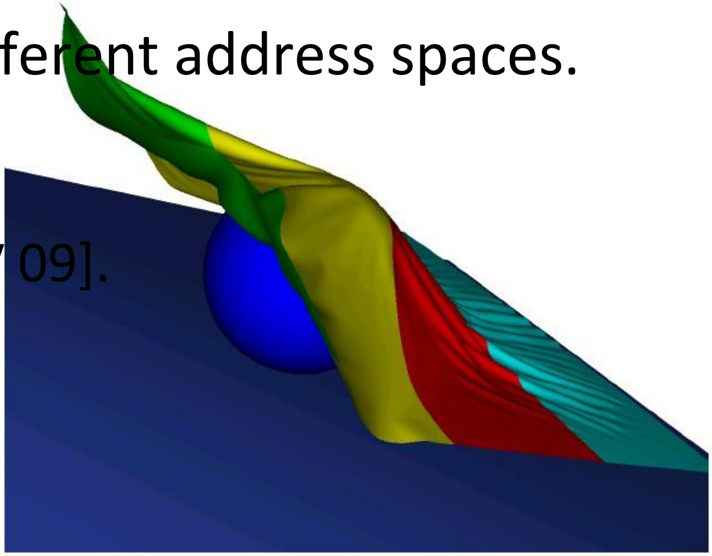
- Program with Divide & Conquer techniques.
- Use MPI_Comm_spawn to (recursively) create new tasks.
  - 1 task == 1 (MPI) process.

- Make sure that the children tasks may communicate with their parent
  - Have the parent send the children input data, and then block.
  - Have the children send their results back to the parent.
- This implies very large-grained parallelism, but at least:
  - You can benefit from dynamic resources.
  - You can improve the load balance.

Tasks & Heterogeneous Parallel Programming

- Adaptive Work Stealing already uses 2 algorithms
  - 1 sequential, 1 parallel.
- Why not using **2 different implementations** (methods) to run on a container, e.g. **one for CPU and one for GPU**?
  - The merge() method handles the different address spaces.
- This has been done partially

  by B. Raffin and E. Hermann [EGPGV 09].
- J. Lima´PhD
  - V. Danjean, T. Gautier, N. Maillard



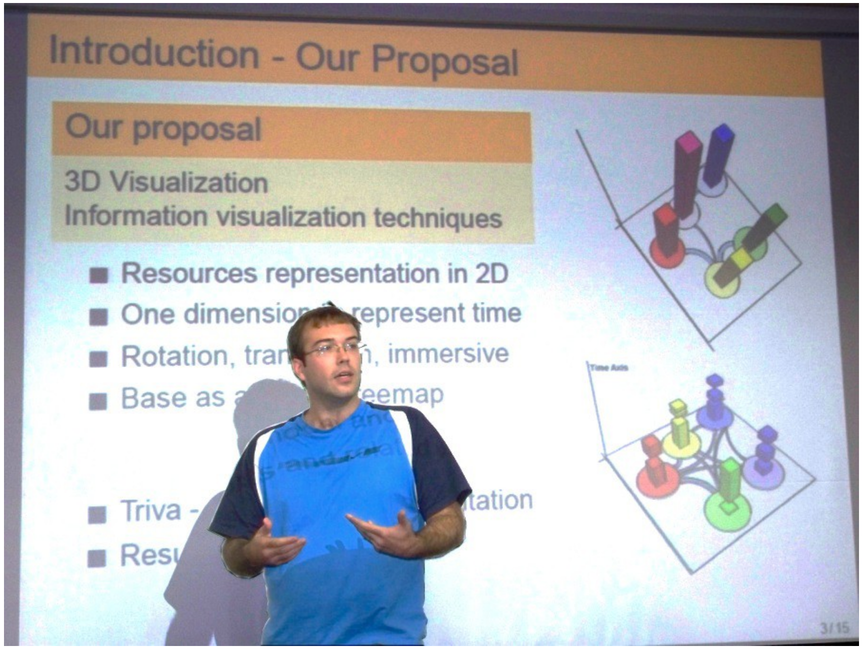
GPPD Grupo de Processamento Paralelo e Distribuído

INSTITUTO DE INFORMÁTICA UFRGS

Monitoring parallel applications

- Using 3D representations to visualize the behavior of parallel programs
  - Resources (x, y) vs. Time (z)
  - Rotations, zooms...

Lucas Schnorr



Introduction - Our Proposal

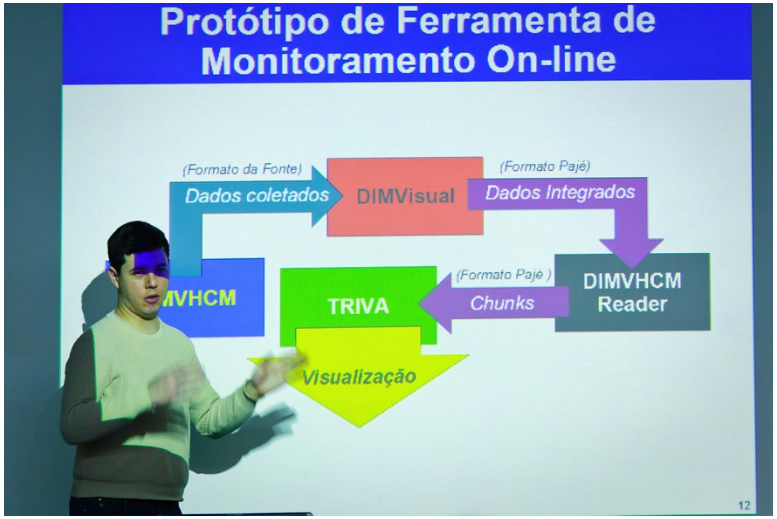
Our proposal

3D Visualization
Information visualization techniques

- Resources representation in 2D
- One dimensio represent time
- Rotation, tran n, immersive
- Base as a eemap
- Triva - tation
- Resu

3/15

Rafael Tesser



Protótipo de Ferramenta de Monitoramento On-line

(Formato da Fonte)
Dados coletados
DIMVisual
(Formato Pajé)
Dados Integrados
MVHCM
DIMVHCM Reader
(Formato Pajé)
Chunks
TRIVA
Visualização

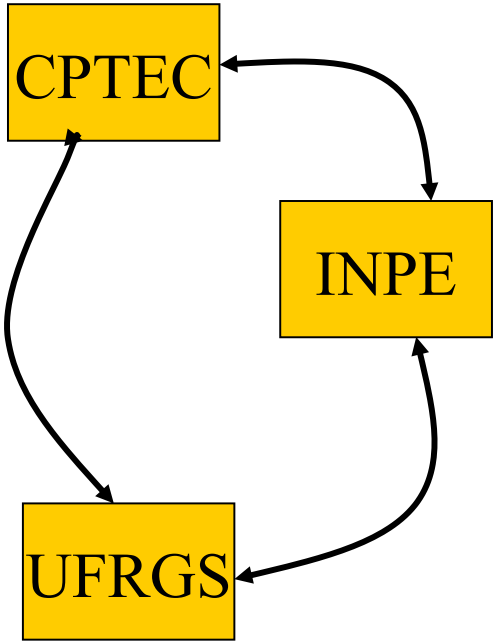
12

Cloud Computing

- Question:
  - Is cloud computing appropriate to execute HPC applications?
- Experimental Platform: Microsoft Azure (PaaS and IaaS)
- People:
  - Eduardo Roloff (graduate student, MsC thesis,
  - Advisor: Philippe Navaux; co-advisor: Alex Carissimi
- Goals and methodology
  - Evaluate Azure PaaS and IaaS solutions on MPI and OpenMP applications
  - NAS benchmarks, Climate applications (BRAMs)
  - Analysis: execution time and system resources

Climate Modeling

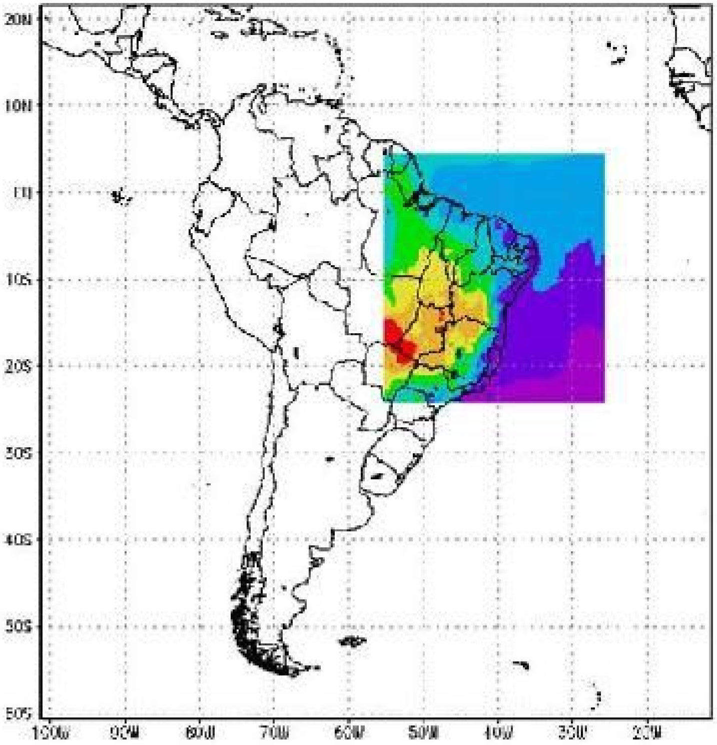
- Joint project with CPTEC, Brazilian National Weather Forecast



CPTEC

INPE

UFRGS



Atmosfera Massiva Project

Team:

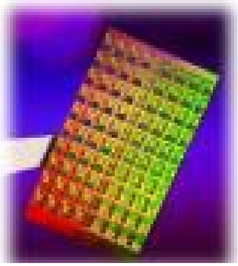
- Distributed and Parallel Processing Group – INF-UFRGS.
- Weather Forecasting and Climate Research Center – INPE.
- National Laboratory of Scientific Computing (LNCC).

Study Topic:

- Meteorological and environmental models.

Objectives:

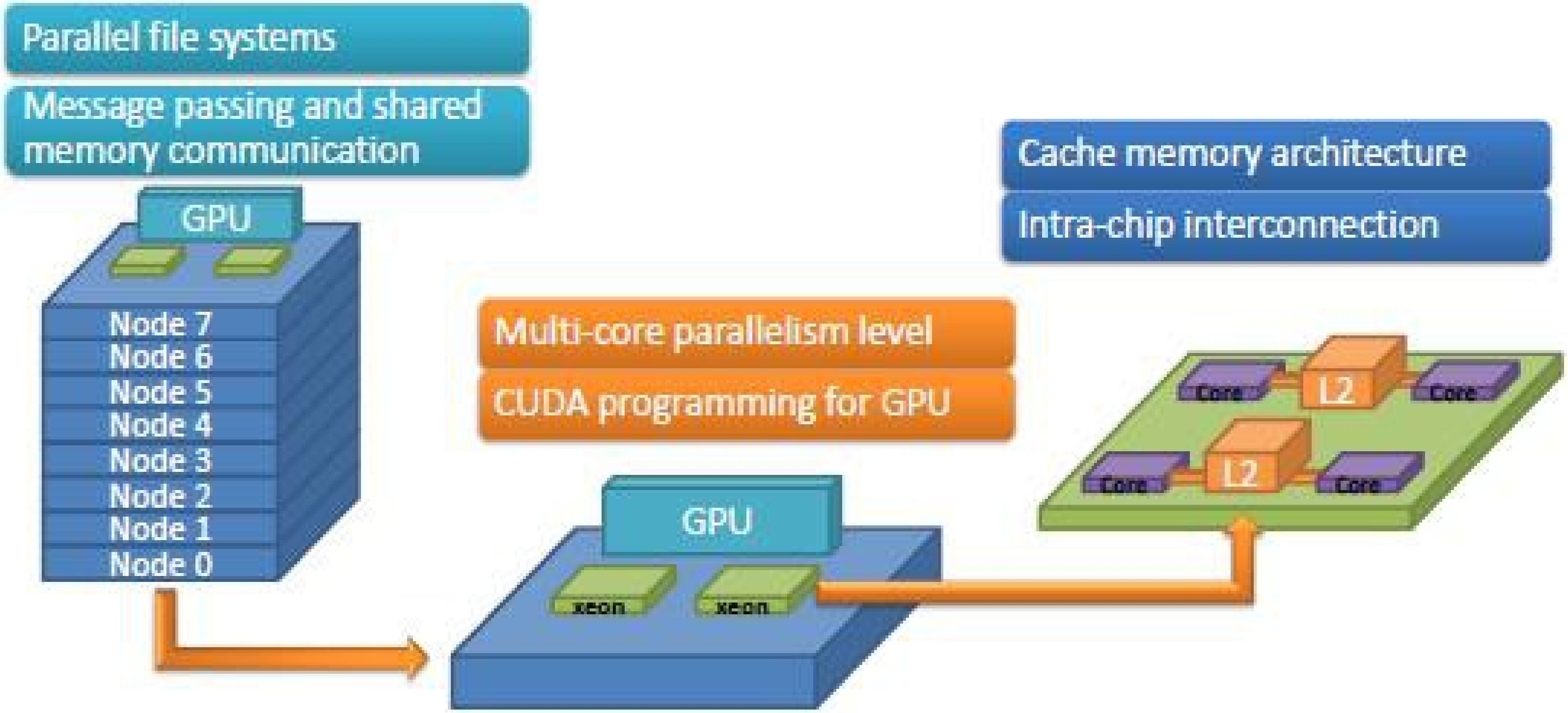
- Study on the impact of multi-core architectures and multi-level parallelism for meteorological and environmental models.



Atmosfera Massiva Project

Challenge:

- Achieve better application's parallelism expression in order to use all the potential from new multi-core architectures.



Parallel file systems

Message passing and shared memory communication

Cache memory architecture

Intra-chip interconnection

Multi-core parallelism level

CUDA programming for GPU

HPC-GA Project

High Performance Computing for Geophysics Applications

Partners: INRIA Grenoble, Bordeaux, Pau

BCAM, BRGM, UFRGS, UNAM, UJF

The HPC-GA project is gathering an international, pluridisciplinary consortium of leading European and South American researchers featuring complementary expertise to face the challenge of designing high performance geophysics simulations for parallel architectures.

Load Balancing Strategies

Motivation

Climate and Weather Models

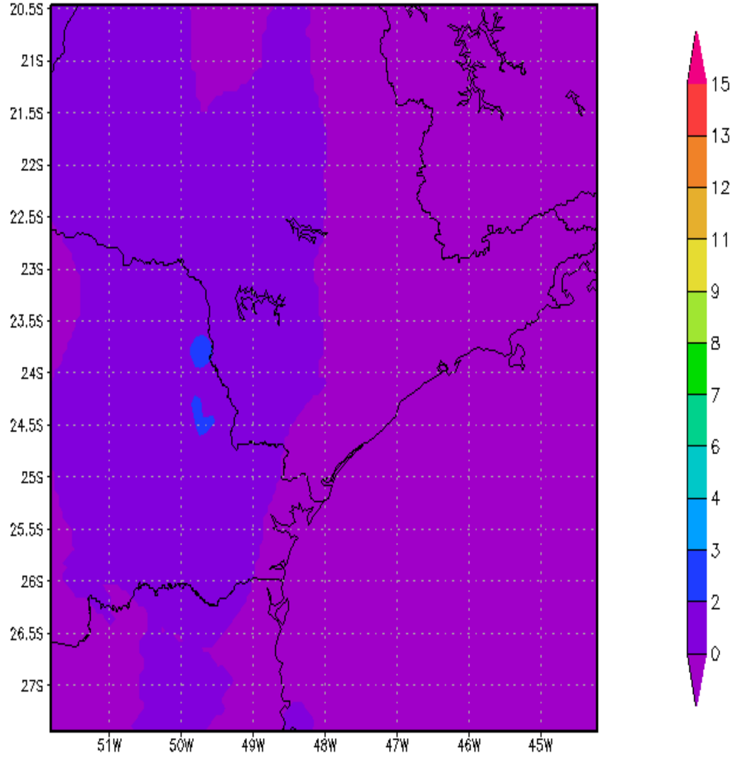
- High computational demands
- Typically run on large supercomputers

Obstacles to Scalability

- Lack of sufficient parallelism
- Load imbalances
  - Static sources: day/night cycle, topography
  - Dynamic sources: atmospheric phenomena
  - Typical solution: changes in model's code
    - Requires intimate knowledge of the application
    - Needs to be redone for each source of imbalance

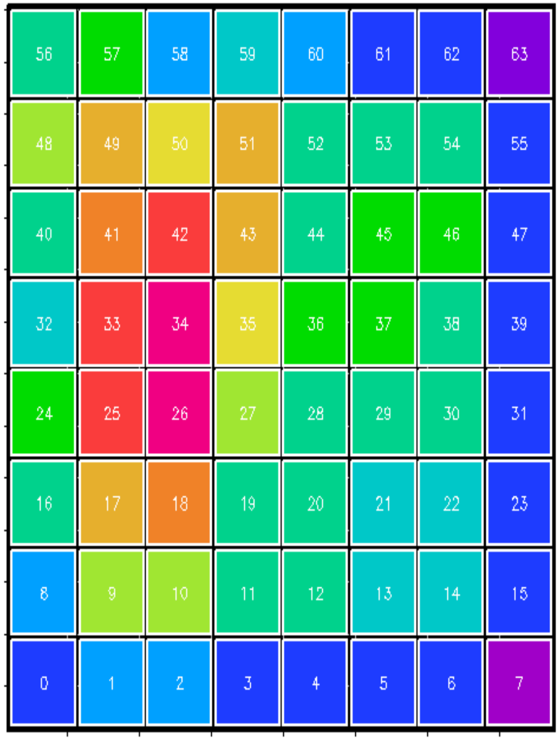
Example of Imbalance

Weather Forecast, Feb.2010



GrADS: COLA/IGES 2010-02-18-09:46

Processor Load, P=64 (8x8)



56	57	58	59	60	61	62	63
48	49	50	51	52	53	54	55
40	41	42	43	44	45	46	47
32	33	34	35	36	37	38	39
24	25	26	27	28	29	30	31
16	17	18	19	20	21	22	23
8	9	10	11	12	13	14	15
0	1	2	3	4	5	6	7

GrADS: COLA/IGES 2010-02-18-10:00

Dynamic Load Balancing for Weather Models via AMPI

Eduardo R. Rodrigues

IBM Research – Brazil

edrodri@br.ibm.com

Celso L. Mendes

University of Illinois – USA

cmendes@illinois.edu

Philippe Navaux

Univ.Fed.RGS - Brazil

navaux@inf.ufrgs.br

Laxmikant Kale

University of Illinois – USA

kale@illinois.edu

Jairo Panetta

CPTEC/INPE - Brazil

panetta@cptec.inpe.br

Approach on Charm++

Leverage Charm++ Run-Time System

- Support for MPI applications via Adaptive-MPI
- Minimal changes required to original MPI code
- Some of the changes can be automated

Charm++ Load Balancing Framework

- Explores migration capability in Charm++
- Balancing policies based on observed load and/or communication traffic
- Same balancers can be applied to different codes
- Various balancing policies available
- Easy to create/code new policies
- Balancers are oblivious to application code details

Case-Study: BRAMS Weather Model

BRAMS Roots

- RAMS, from Colorado State University
- Model adapted for the tropics
- Software structure modernized at CPTEC-Brazil

Major BRAMS Features

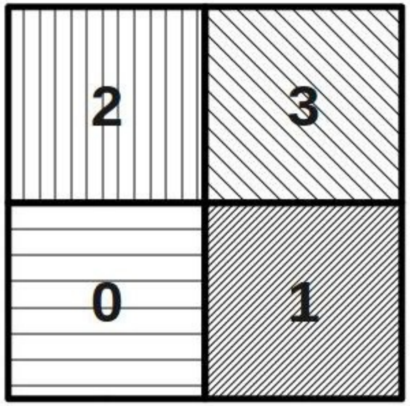
- Fortran90 + MPI parallelization
- Open source, many thousands of lines of code
- Research and production versions available
- Support for multiple nested grids
- Recently extended with a coupled aerosol and tracer transport model (CATT-BRAMS)
- Daily production use, across Brazil and abroad

Load Balance Approach

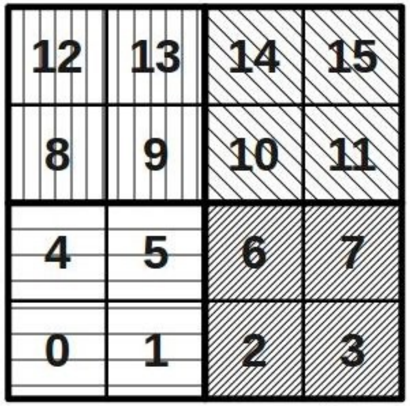
Adaptive MPI (AMPI):

- MPI implementation based on Charm++
- Transforms MPI ranks into *user-level* threads
- Processor Virtualization:
    - Multiple threads (MPI ranks) per processor
    - Mapping controlled by the Charm++ runtime
    - Example: MPI code on 4 processors

MPI:



AMPI:

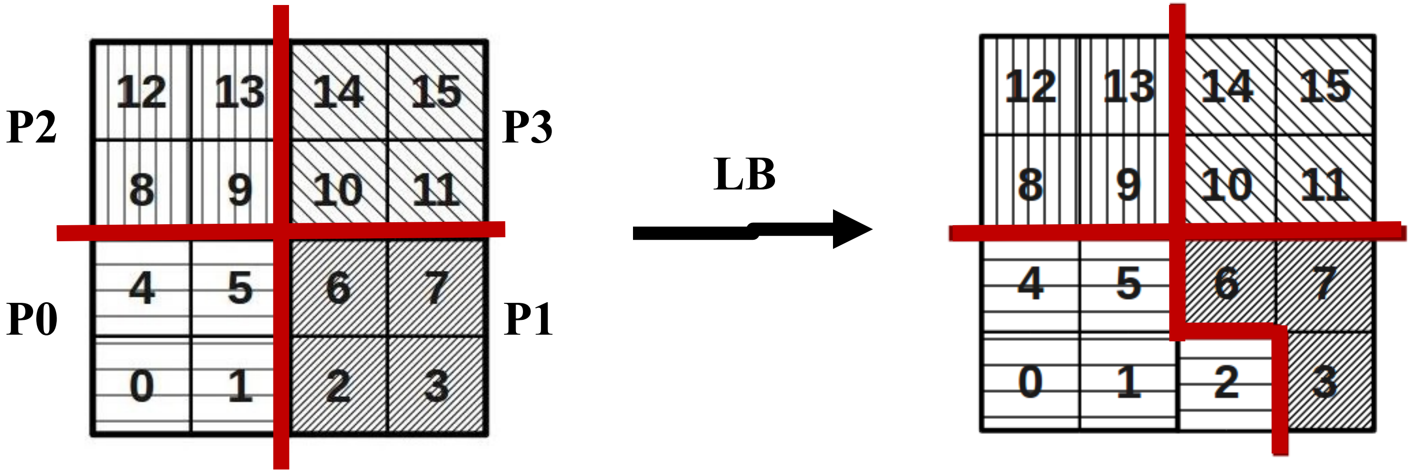


Load Balance via AMPI

Key AMPI Feature: Migration

May migrate threads across processors during execution to enable load balancing (LB)

Example: migration of thread 2 from P1 to P0



Dynamic, measurement-based load balancing

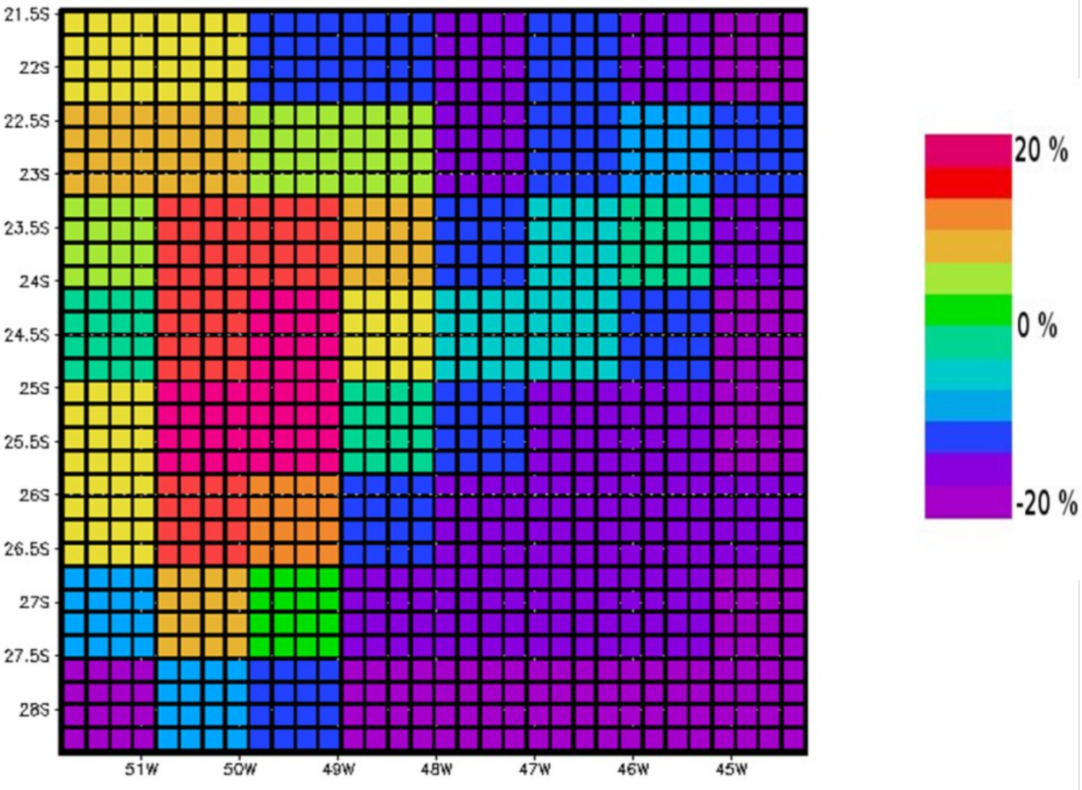
Measurements can be turned on/off in given phases

Balancing/migration points can be selected by user

AMPI’s Load Balance in BRAMS

Load distribution after 600 tsteps

P=64, 1024 AMPI threads (=1024 MPI ranks)

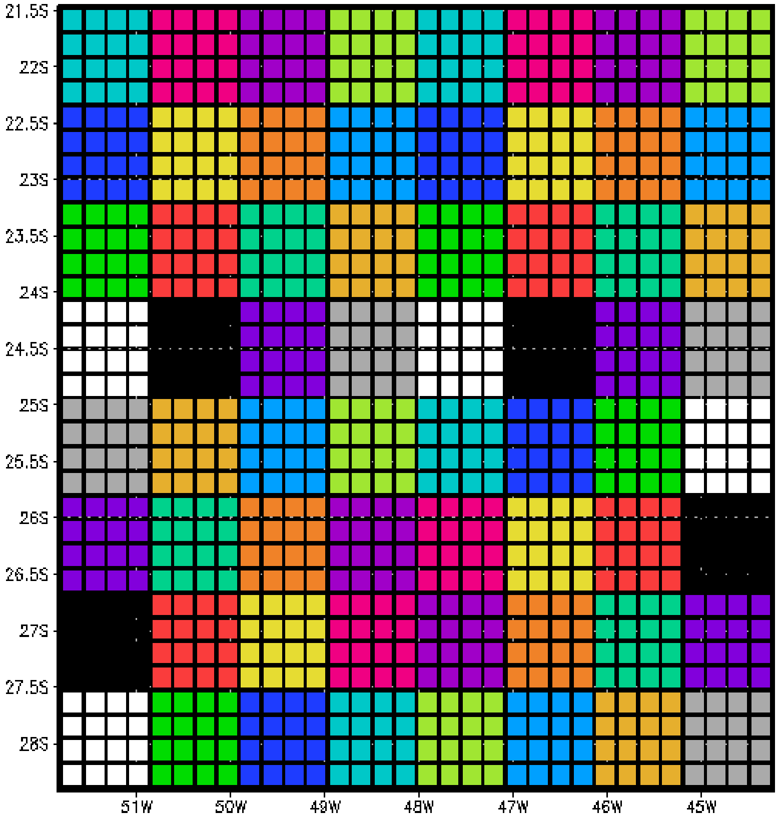


AMPI’s Load Balance in BRAMS

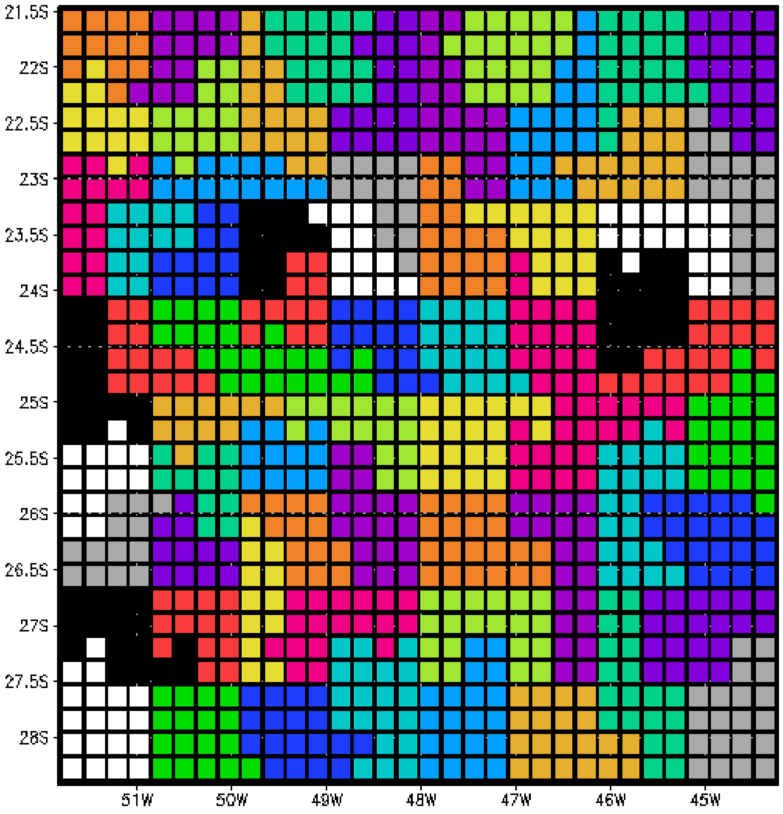
Mapping of Threads Before/After LB

(Random colors used here, for illustration only)

initial mapping



mapping after LB

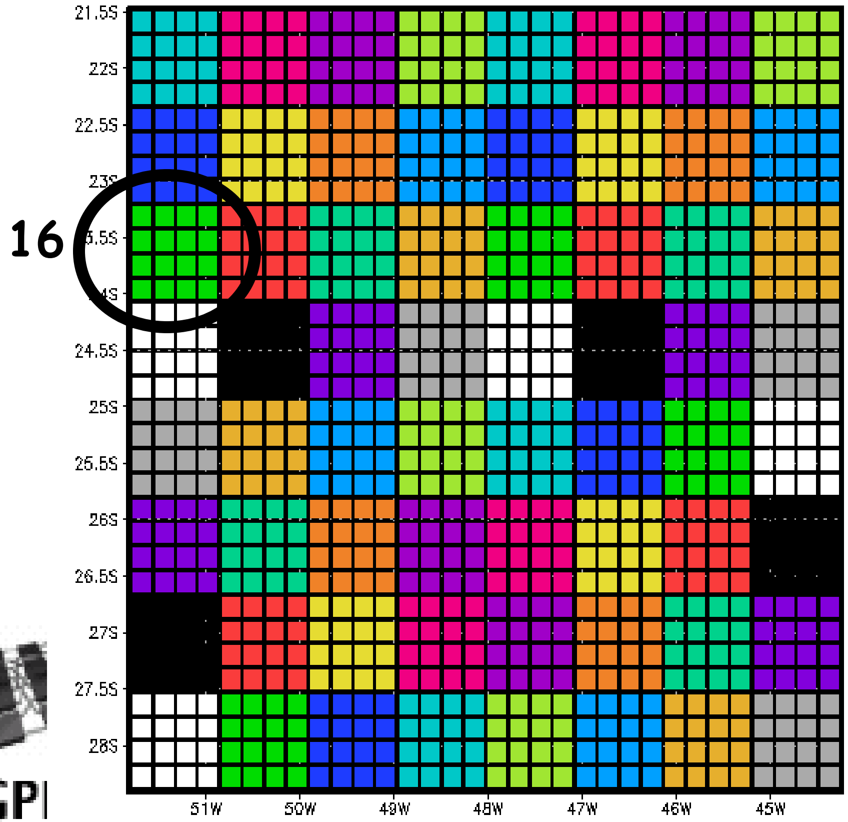


AMPI’s Load Balance in BRAMS

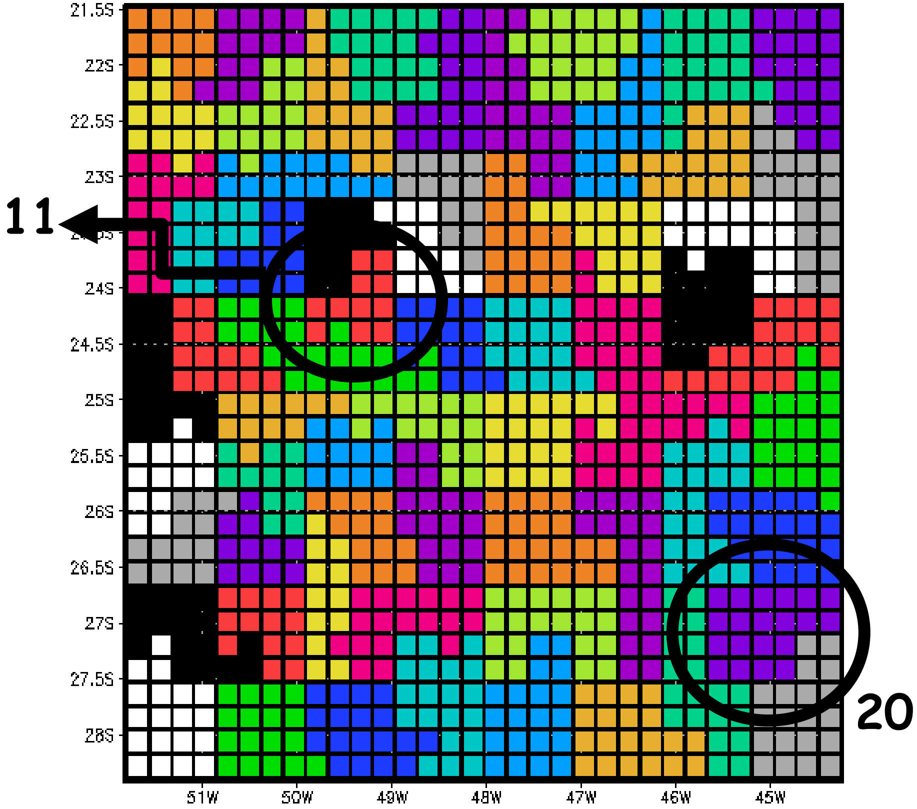
LB Leads to Variable #Threads/Proc

Such that **load** is uniform across processors

initial mapping



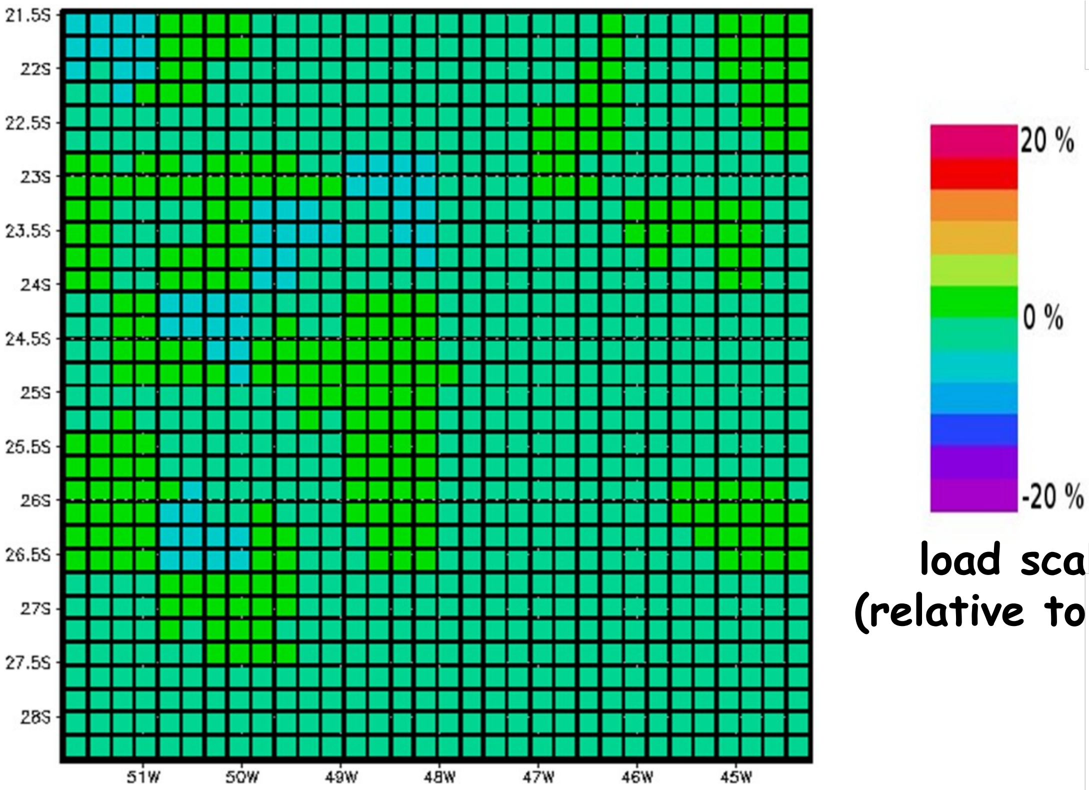
mapping *after* LB



AMPI’s Load Balance in BRAMS

Load distribution after Load Balance

Hilbert-LB balancer used (good for 2D domains)



load scale (relative to average)

Another case: Port Ondes3D to AMPI

Virtual processors support:

Removal of global an static variables;

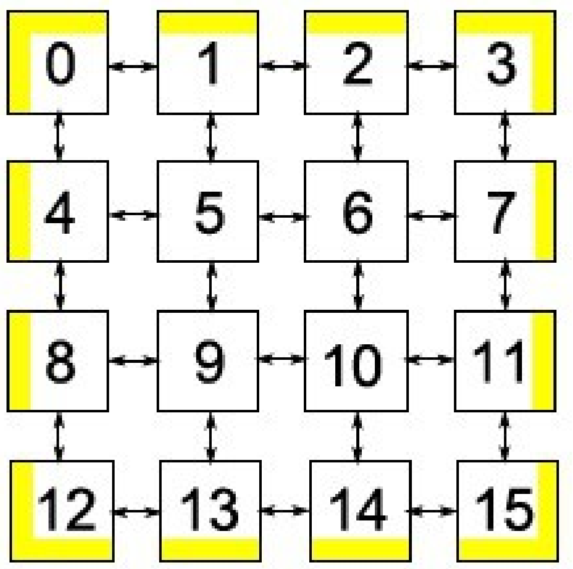
- due to the use of user-level threads in place of processes;

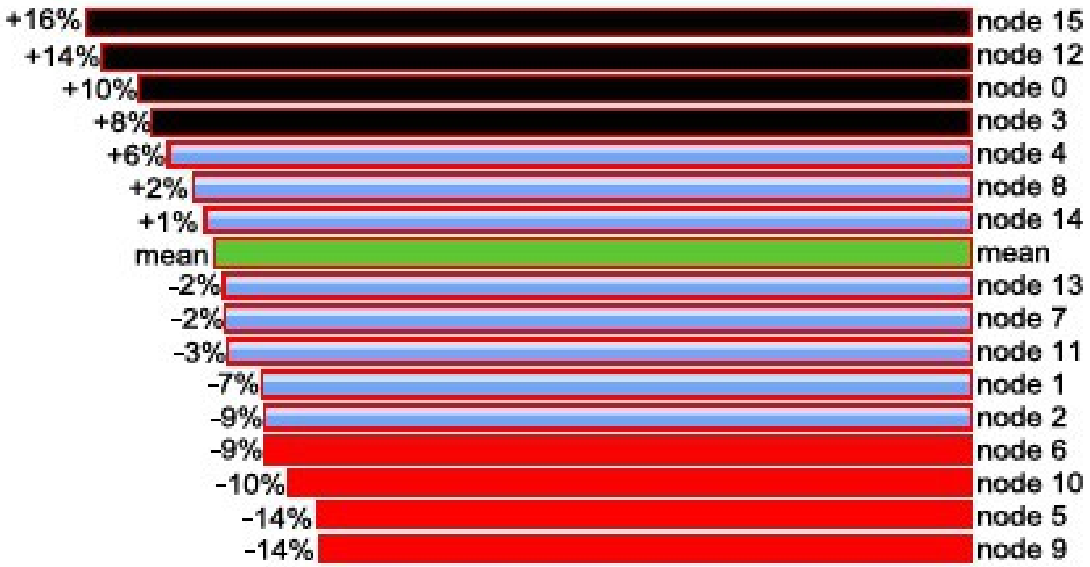
Support to process migration:

- Implementation of functions for data serialization;
  - PUP functions: **P**acking and **U**npacking;

Destruction and creation of MPI_Request variables.

Ondes 3D imbalance





Thank you

navaux @ inf.ufrgs.br

http://gppd.inf.ufrgs.br/