A Steering Environment for Online Parallel Visualization of Legacy Parallel Simulations

Aurélien Esnard, Nicolas Richart and Olivier Coulaud

ACI GRID EPSN (French Ministry of Research Initiative)
ScAlApplix Project at INRIA Futurs
LaBRI and University of Bordeaux 1

DS-RT 2006
Torremolinos, Malaga, Spain.
October 2-4, 2006.

Outline

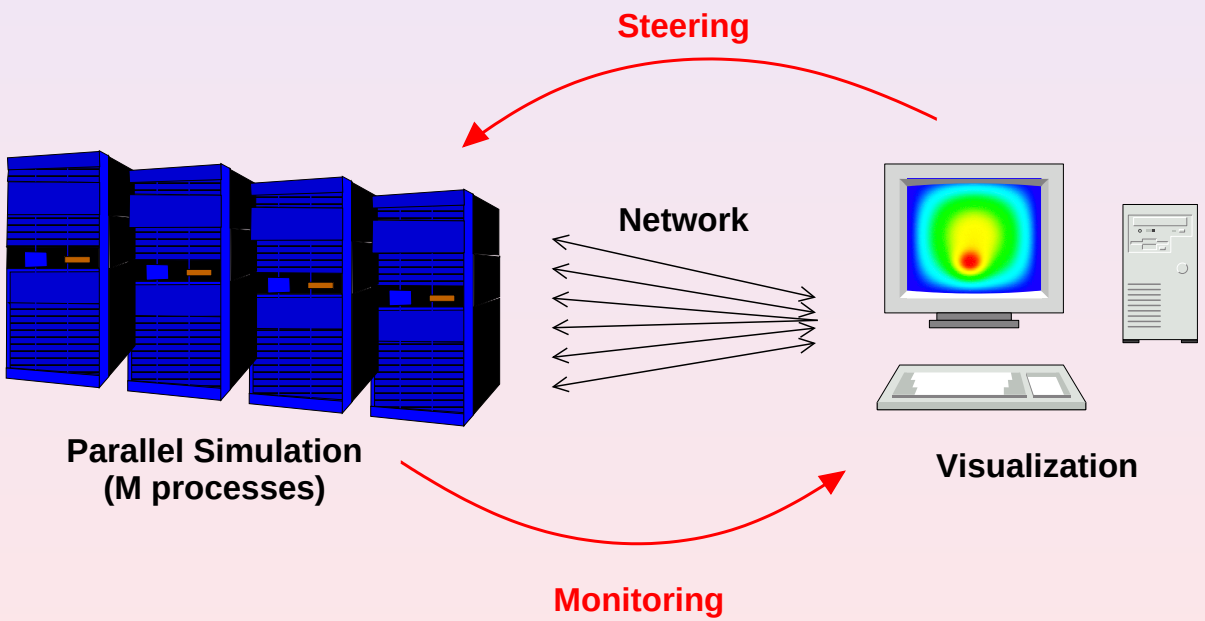
1. Introduction
   - Computational Steering
   - Related Works
2. The EPSN Framework
   - Overview & Architecture
   - Redistribution Algorithm for Unstructured Data
   - Online Parallel Visualization with EPSN
3. Results
   - Case Study: The Gadget2 Cosmological Simulation

Outline

1. Introduction
   - Computational Steering
   - Related Works
2. The EPSN Framework
   - Overview & Architecture
   - Redistribution Algorithm for Unstructured Data
   - Online Parallel Visualization with EPSN
3. Results
   - Case Study: The Gadget2 Cosmological Simulation

Numerical Simulations and Computational Steering

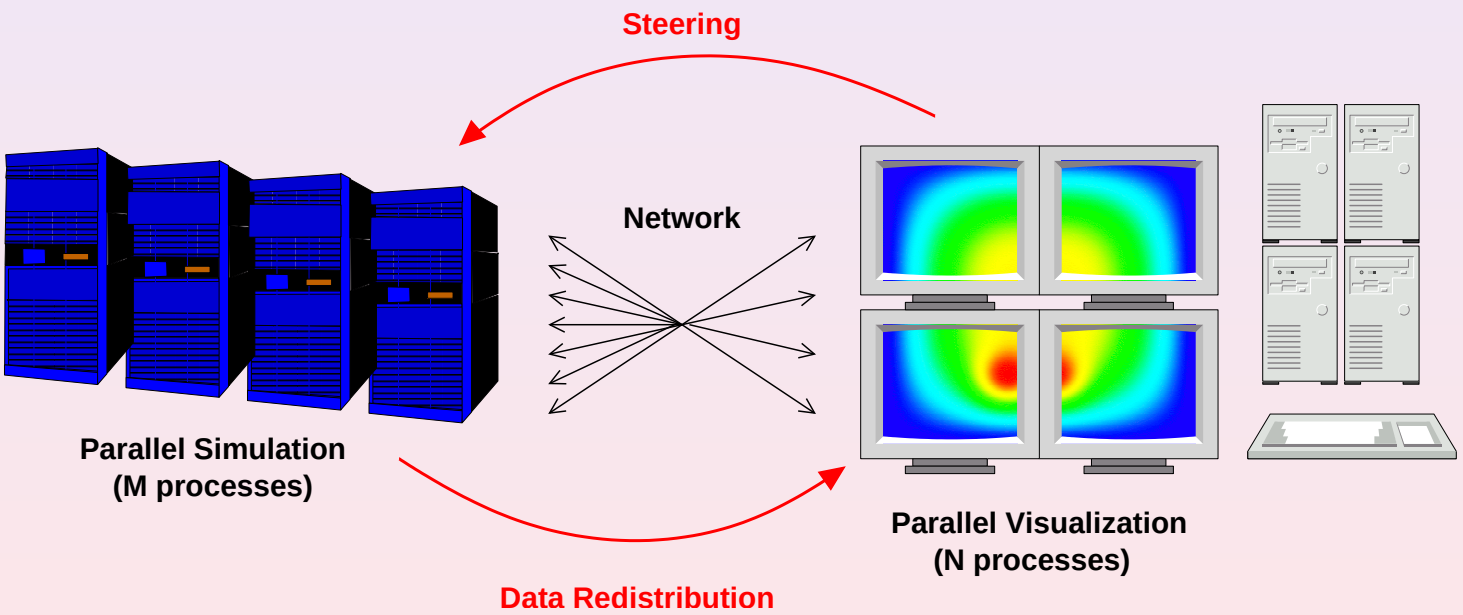
- Visualization as post-processing step (batch mode)
  $\rightarrow$ tedious, lack of control over the in-progress computations
- Computational steering as a more interactive approach
  $\rightarrow$ coupling simulation and visualization through the network
- Online visualization of intermediate results (monitoring)
- Change simulation parameters or data on-the-fly (steering)



$\Rightarrow$ **Drive the simulation more rapidly in the right-direction**

M × N Computational Steering

- Online visualization requires performance
  → process large and complex datasets, display results with high-resolution, ...
- To avoid the bottleneck of sequential visualization
  → idea: use of parallelism for both the simulation (M) and the visualization (N)
- An attractive approach both in terms of cost and performance
  → parallel visualization and parallel rendering with a PC-based graphics cluster



⇒ **It raises the difficult problem of parallel data redistribution...**

Outline

1. Introduction
   - Computational Steering
   - Related Works
2. The EPSN Framework
   - Overview & Architecture
   - Redistribution Algorithm for Unstructured Data
   - Online Parallel Visualization with EPSN
3. Results
   - Case Study: The Gadget2 Cosmological Simulation

Related Works

Most of steering environments (CUMULVS, DAQV, ...)

- support parallel simulations (shared-memory, distributed-memory)
- but only with sequential visualization systems

Some recent works

- gViz: distributed modules (simulation, visualization, rendering)
  $\longrightarrow$ visualization and rendering modules are still sequential
- SCIRun/Uintah PSE: parallel visualization module
  $\longrightarrow$ but only running in shared-memory (no redistribution problem)

Our steering environment, EPSN

- Steering of parallel simulations with parallel visualization tools in distributed-memory (M$\times$N computational steering)

Outline

1. Introduction
   - Computational Steering
   - Related Works
2. The EPSN Framework
   - Overview & Architecture
   - Redistribution Algorithm for Unstructured Data
   - Online Parallel Visualization with EPSN
3. Results
   - Case Study: The Gadget2 Cosmological Simulation

Overview of the EPSN Framework

A software environment for M×N computational steering

- Legacy parallel simulations (C, C++ or Fortran)
- Sequential or parallel visualization program

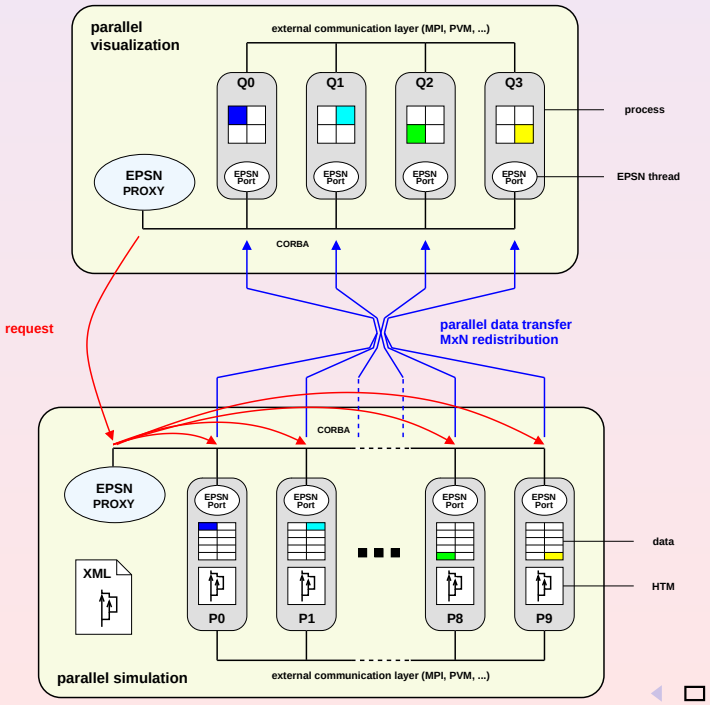
Integration of legacy simulations

- Source-code annotations with the EPSN back-end API
- Abstract model to describe the simulation
  → Description of its control-flow: Hierarchical Task Model (HTM)
  → Description of its data: complex objects (grids, particles, meshes)

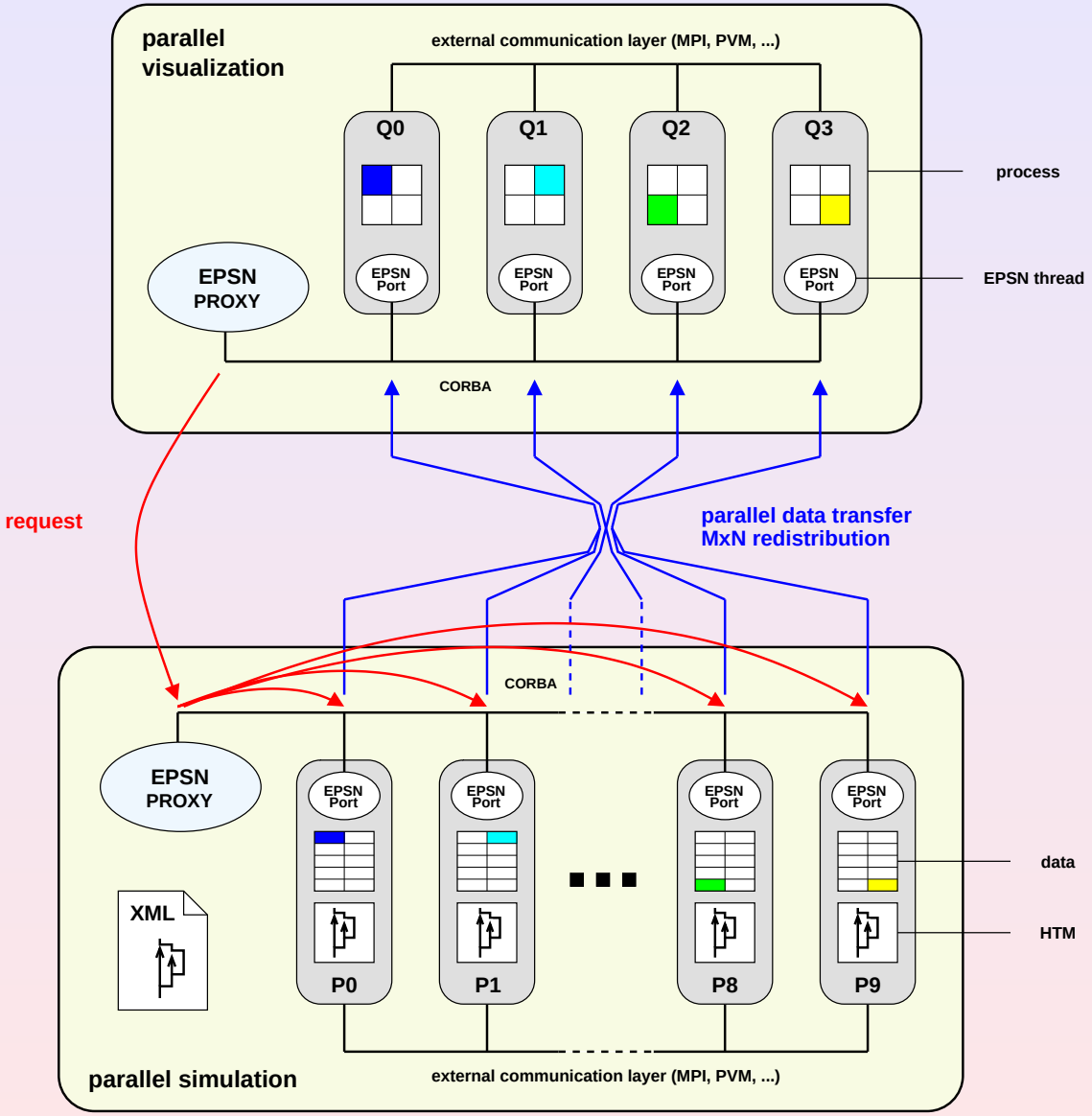
⇒ **This model intends to clarify where, when and how one can safely interact with the simulation**

Architecture of the EPSN Framework

- Client/server relationship (simulation = server ; visualization = client)
- Dynamic and distributed infrastructure
  → several clients can connect and disconnect a remote simulation on-the-fly
- Communication infrastructure based on CORBA
  → CORBA server running on each node + proxy
- Steering of the simulation is based on requests sent by clients
  → control (play, pause), data access (get, put), action



Architecture of the EPSN Framework



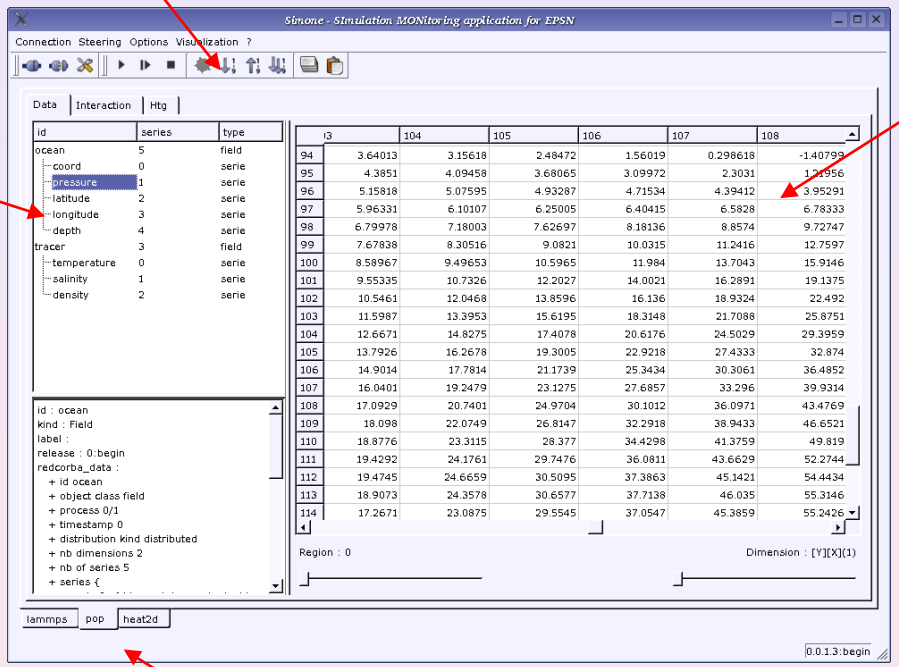
Simone (Simulation Monitoring Interface for EPSN)

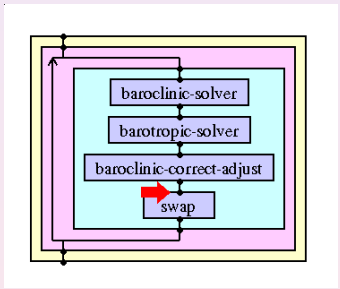
A generic user interface for EPSN to easily interact with your simulation

Request Panel (Control and Data Access)

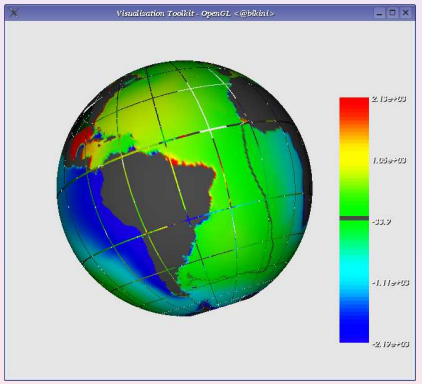
List of Simulation Data

Data Sheet





Hierarchical Task Model



Visualisation Plugins

List of Connected Simulations

Current Date

Simone connected to the Parallel Ocean Program (POP) of the LANL

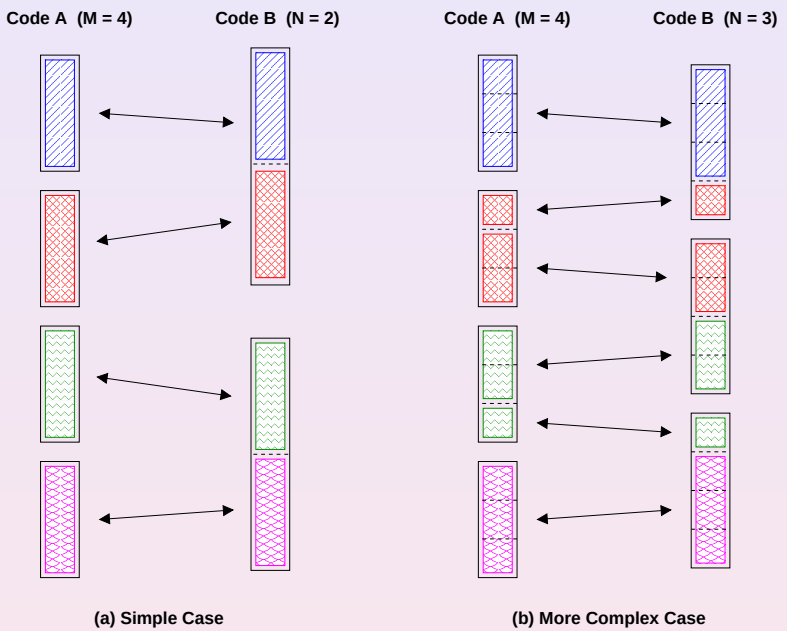
Outline

1. Introduction
   - Computational Steering
   - Related Works
2. The EPSN Framework
   - Overview & Architecture
   - Redistribution Algorithm for Unstructured Data
   - Online Parallel Visualization with EPSN
3. Results
   - Case Study: The Gadget2 Cosmological Simulation

Redistribution Layer: Placement Approach

A redistribution algorithm that is well adapted to the context of M×N computational steering

- $M \neq N$, different data distributions between codes
- Data distribution not initially defined on the visualization side
- The redistribution layer can choose it at run-time “in the best way”
- Placement problem of the simulation elements to the N visualization processes
- Message generation requires to define a split operator for the object you consider

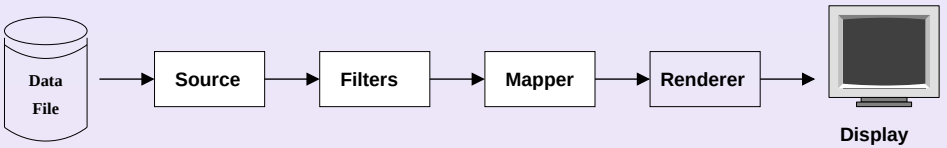


A very generic approach that is used by EPSN for different kinds of objects (structured grids, particles, unstructured meshes)

Outline

1. Introduction
   - Computational Steering
   - Related Works
2. The EPSN Framework
   - Overview & Architecture
   - Redistribution Algorithm for Unstructured Data
   - Online Parallel Visualization with EPSN
3. Results
   - Case Study: The Gadget2 Cosmological Simulation

The classical visualization pipeline

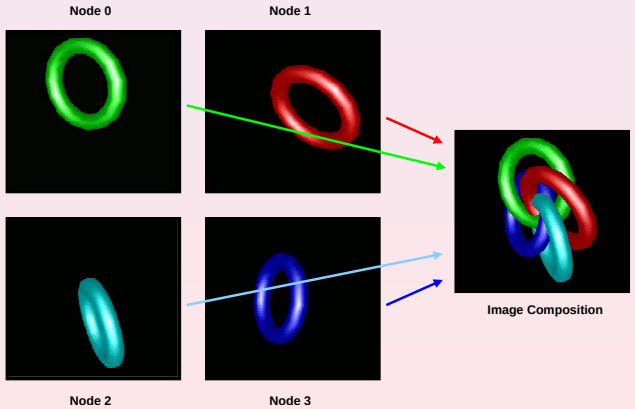


Parallel visualization

→ the pipeline is fully replicated on each node of the graphics cluster and data are distributed on these nodes

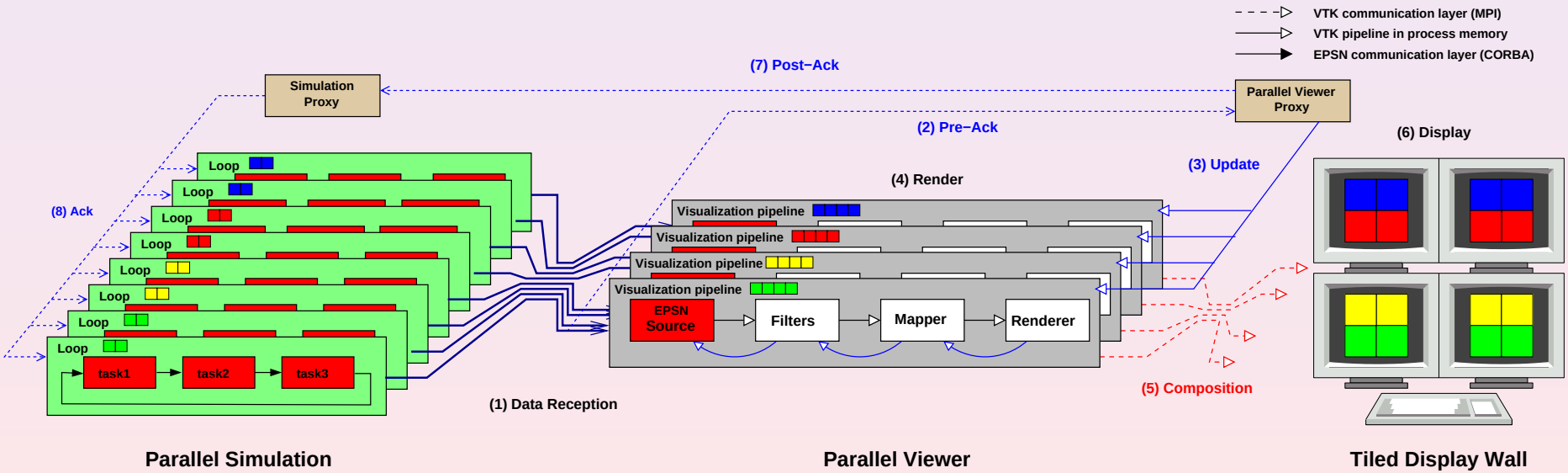
Parallel rendering techniques

→ enable to combine the capabilities of several graphics cards to produce the final image (e.g. sort-last algorithm)



The different steps required to perform online parallel visualization with EPSN:

- Data reception by EPSN sources (step 1)
- Pipeline update request (steps 2-3)
- Parallel rendering (step 4), image composition (step 5) and display (step 6)
- Acknowledgement of the simulation to signal the image update (steps 7-8)



The EPSN Software

EPSN framework

- 2 libraries: simulation side & visualization side
- Written in C++ (bindings for C/Fortran codes)
- OmniORB4: high-performance implementation of CORBA
- The redistribution layer is packaged in an independant library called RedGRID

Parallel viewer

- Parallel visualization based on VTK (Visualization ToolKit)
- Parallel rendering on TDW thanks to Ice-T library developed at Sandia

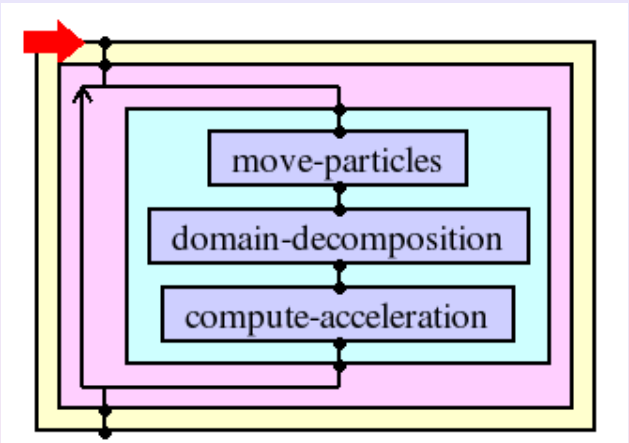
RedGRID and EPSN are available at INRIA Gforge (LGPL)
→ `redgrid.gforge.inria.fr`
→ `epsn.gforge.inria.fr`, `www.labri.fr/epsn`

Outline

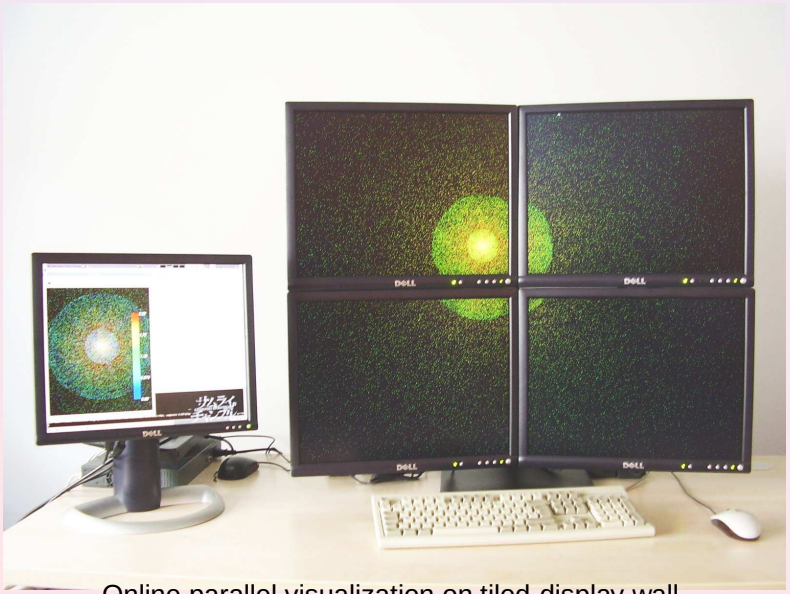
1. Introduction
   - Computational Steering
   - Related Works
2. The EPSN Framework
   - Overview & Architecture
   - Redistribution Algorithm for Unstructured Data
   - Online Parallel Visualization with EPSN
3. Results
   - Case Study: The Gadget2 Cosmological Simulation

Case Study: The Gadget2 Cosmological Simulation

- Parallel legacy code written in C and using MPI (Message Passing Interface)
- Developed by V. Springel at Max-Plank Institute of Astrophysics
- Simulates birth of a galaxy that collapses gravitationally until a central shock
- Galaxy represented by a gas cloud (1,000,000 particles distributed on 60 processes for our test case)
- Astrophysicists want to visualize the evolution of the galaxy in 3D



Representation of the Gadget2 simulation in EPSN.



Online parallel visualization on tiled-display wall.

Gadget2 average time for simulation computations, data transfer and visualization (in ms/iteration)

- No overhead for the simulation with EPSN, without visualization
- Huge overhead in the sequential visualization case (+21%)
- Very small overhead in the parallel visualization case (+2%)
- Huge overhead for higher global resolution (network bandwidth not adapted!)

M	N	S	Global Resolution	Transfert Time	Visualiz. Time	Simulation Time	
						Total	Overhead
60	–	–	–	–	–	2150	–
60	1	1	1600×1200	140	1160	2600	+21%
60	4	1	1600×1200	91	620	2180	+1.4%
60	4	4	1600×1200	91	500	2155	+0.2%
60	4	4	3200×2400	90	1300	2670	+24%

M = number of simulation processes; N = number of visualization processes; S = number of screens.

Conclusion

A modern approach for M×N computational steering

- Parallel visualization and rendering techniques
- Redistribution algorithms based on a placement approach well adpated for computational steering
- Validation with a real-life simulation in astrophysics (Gadget2)

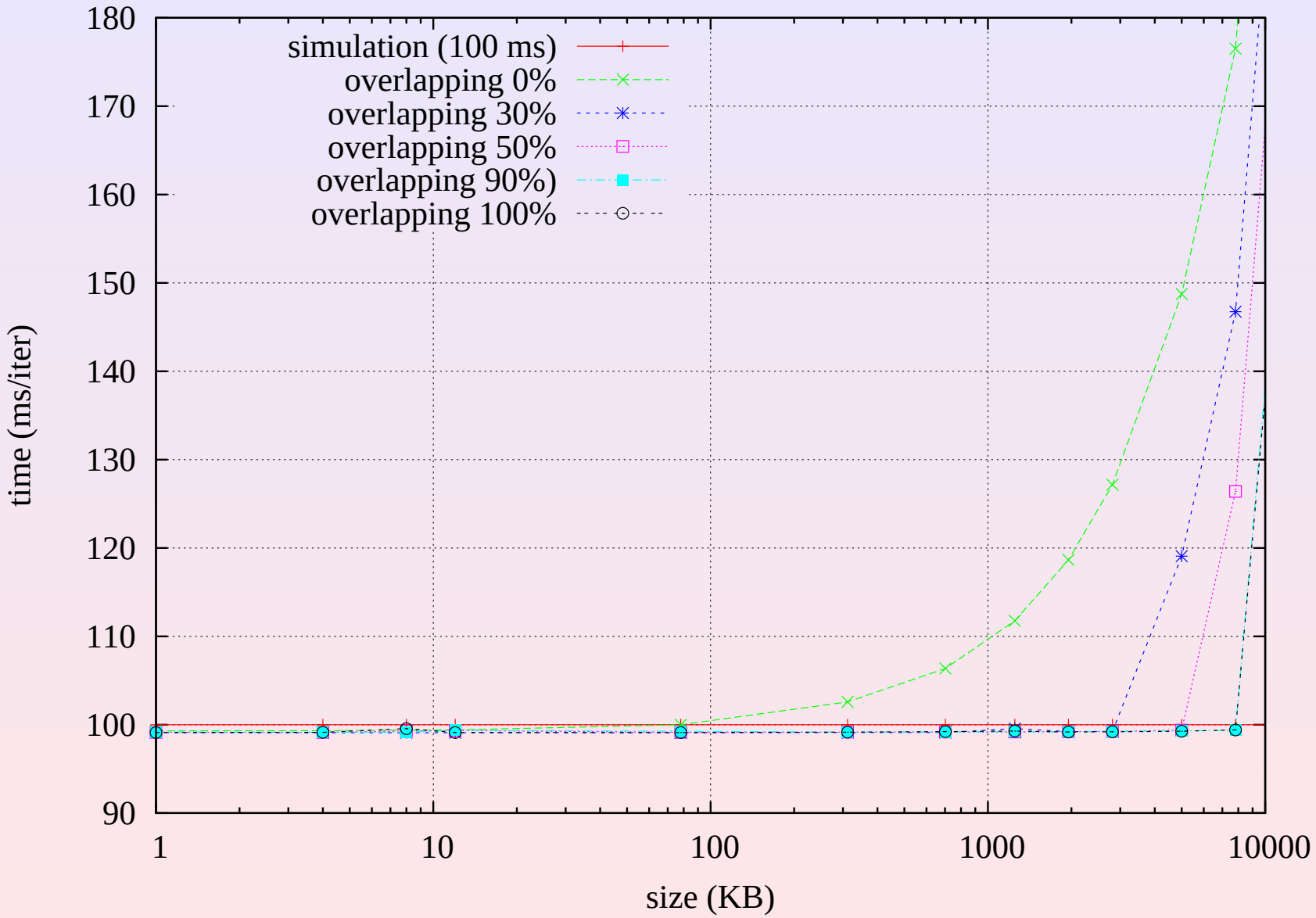
In future works

- Integration of our solution in a high-level visualization system like Paraview
- Redistribution of more complex objects (multi-level grids, AMR, ...)
- Steering of parallel-distributed simulations (e.g. multi-physics)

Outline

4 Appendix
- Performance of the EPSN Framework

Overlapping of the Steering Overhead



Redistribution and Parallel Data Flow