Performance Characterization of the Alya Fluid Dynamics Simulator

Guilherme Antonio Camelo, **Lucas Mello Schnorr**
{gacamelo,schnorr}@inf.ufrgs.br

GPPD/INF/UFRGS, Porto Alegre, Brazil

Porto Alegre, September 2016 — WSPPD 2016

Outline

Context: Numeric Simulation and HPC

- Numeric simulation is used to understand/study a real life scenario
- High performance computers used as execution platforms
- Problems often require high computational power

Context: Numeric Simulation and HPC

- Numeric simulation is used to understand/study a real life scenario
- High performance computers used as execution platforms
- Problems often require high computational power
- Distributed and parallel programming techniques provide high computational power e.g OpenMPI

Context: Numeric Simulation and HPC

- Numeric simulation is used to understand/study a real life scenario
- High performance computers used as execution platforms
- Problems often require high computational power
- Distributed and parallel programming techniques provide high computational power e.g OpenMPI
- OpenMPI is ideal to deal with complex physical problems such as fluid dynamic simulation.

Context: Alya - Fluid Dynamic Simulation tool

MPI-based framework to numerically solve physical problems

Context: Alya - Fluid Dynamic Simulation tool

MPI-based framework to numerically solve physical problems

- Simulates fluid dynamic scenarios
- Open source project
- Parallel execution
- Part of the `Prace Benchmark` from BSC (Barcelona Supercomputing Center)

Motivation: Irregular Load

Often numerical simulation applications have irregular load during execution.

Motivation: Irregular Load

Often numerical simulation applications have irregular load during execution.

- Irregular control and data structure
- Adaptive mesh refinement
- Irregular iteraction patterns among processes

Motivation: Irregular Load

Often numerical simulation applications have irregular load during execution.

- Irregular control and data structure
- Adaptive mesh refinement
- Irregular iteraction patterns among processes

Those factors usually lead to load imbalance among both resources and time as the simulation advances

Motivation: Load Imbalance

Finding out application behavior is key to apply optimization techniques such as:

- Better balancing algorithms
- More appropriate communication patterns.

Motivation: Load Imbalance

Finding out application behavior is key to apply optimization techniques such as:

- Better balancing algorithms
- More appropriate communication patterns.

Tracing techniques are used to acquire this information

- Keeps track of information regarding the application
- Record events
- Record MPI operations

The Extrae tracing tool from BSC is used in this experiment

Goal

Characterize Performance of Alya for the case studied

- Identify irregular execution behavior regarding the resources and the time
- Understand Alya's behavior in a smaller scale plataform

Execution Environment

Execution performed on 4 nodes that are part of the Draco cluster at INF UFRGS (Draco has 8 nodes).

Execution Environment

Execution performed on 4 nodes that are part of the Draco cluster at INF UFRGS (Draco has 8 nodes).

Each node equipped with:

- Intel (R) Xeon (R) E5-2640 v2 CPU @ 2.00GHz processor
- 16 physical cores (32 with hyperthreading)
- 64 gigabytes of RAM
- Ubuntu 14.04.4 LTS

Methodology

After experimenting with various configurations, only 4 node and 64 cores were used in the experiment, running only until the 3rd timesteps.

Methodology

After experimenting with various configurations, only 4 node and 64 cores were used in the experiment, running only until the 3rd timesteps.

Trace size 2,7 Gb.

Files generated by the tracing tool Extrae are too big.
Thus, reducing the number of timesteps was necessary.

Methodology

Execution: OpenMPI 1.10.2

- Outputs simulation result

Tracing: Extrae 3.3.0

- Creates individual traces for each processor
- Record communication and execution information
- Trace files created in `.mpits` format

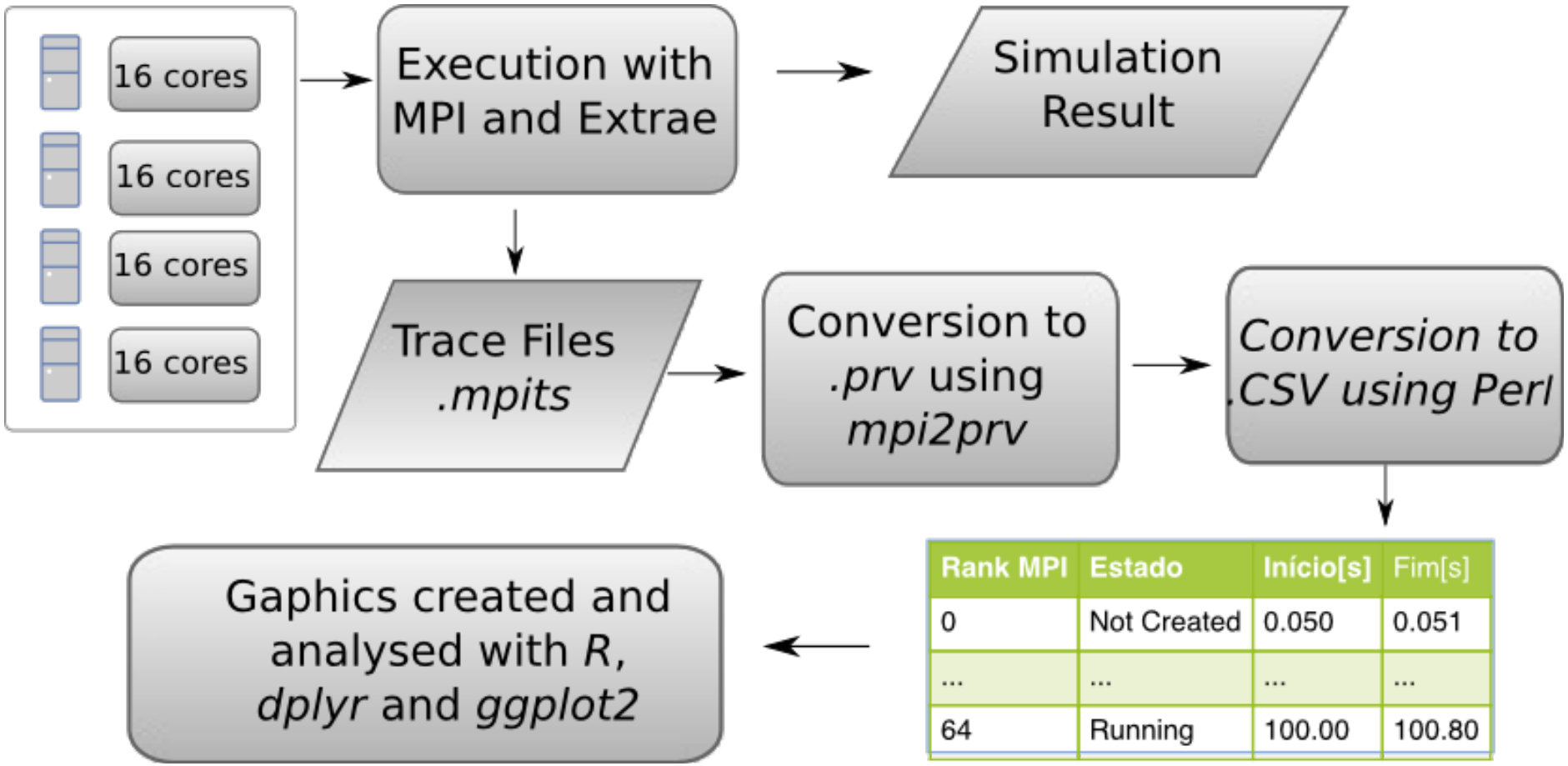
Converting:

- Trace files are converted to `.prv` format with `mpi2prv`
- `.prv` files are converted to `.CSV` (Comma-Separated Values) using a `Perl` script

Plot and Analysis:

- Using `R`, `dlyr` and `ggplot2` different graphics and analysis are made

Methodology

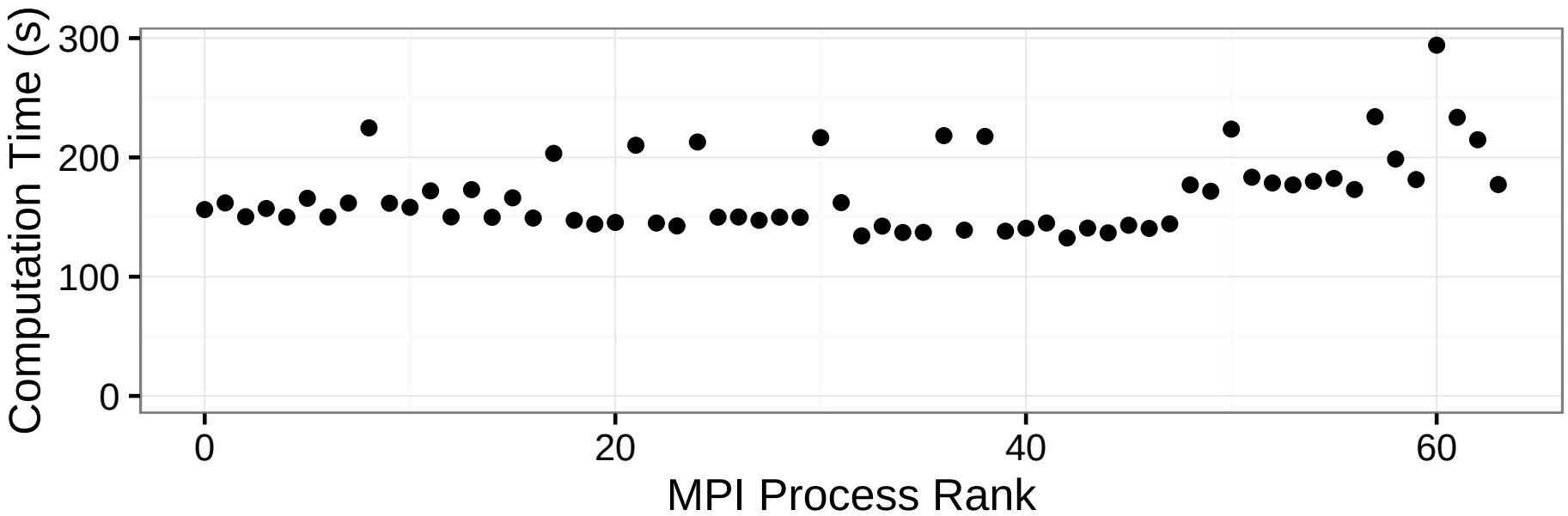


Rank MPI	Estado	Início[s]	Fim[s]
0	Not Created	0.050	0.051
...	...	...	...
64	Running	100.00	100.80

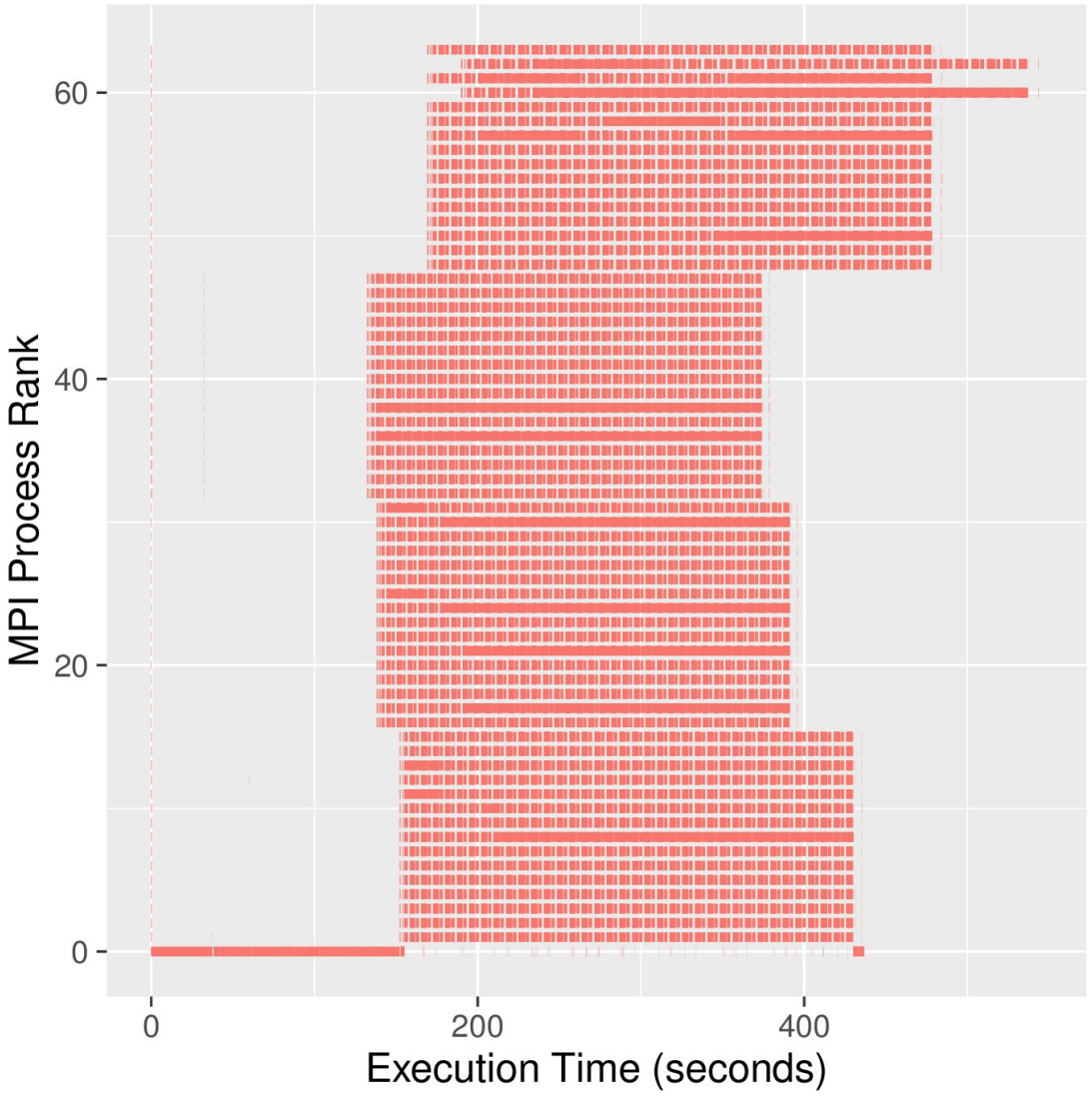
Results

All results are extracted from the same execution with 4 nodes 64 cores and 3 timesteps.

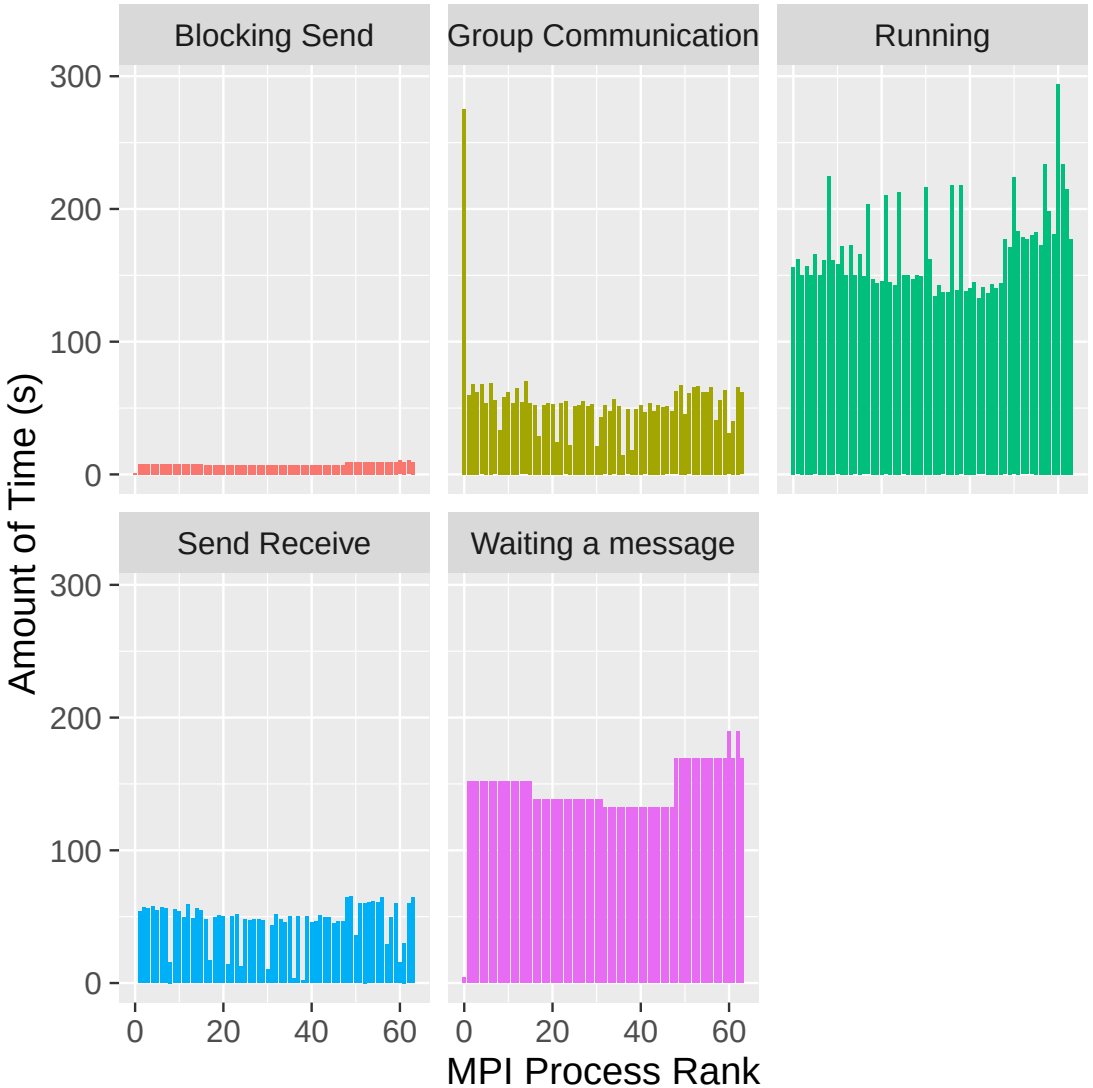
Overview of Load Imbalance



Space/Time Execution



Process Behavior by State



Percent Imbalance

The percent imbalance formula is used to calculate the load balance of applications. It characterizes the uneven distribution of work.

- L_{max} - is the value of the process with the highest load (seconds executing)
- $\overline{L}$ is the average computational load among processes

$$\lambda = (\frac{L_{max}}{\overline{L}} - 1) * 100 \quad (1)$$

The metric is calculated considering the entire execution. One global metric. Load imbalance is 75%, it indicates that if the load were better distributed we could have potential performance improvement

Conclusion

- The execution traced using the Extrae, enabled us to discover relevant information about the core operation of *Alya* in the addressed case

Conclusion

- The execution traced using the Extrae, enabled us to discover relevant information about the core operation of *Alya* in the addressed case

- Significant time wasted in the beginning of the execution probably to divide the problem
- Creates overhead since remaining processes are kept idle

Conclusion

- The execution traced using the Extrae, enabled us to discover relevant information about the core operation of *Alya* in the addressed case

- Significant time wasted in the beginning of the execution probably to divide the problem
- Creates overhead since remaining processes are kept idle

- Anomalies were detected in some processes
- Root process (zero) sends the partitioned data sequentially to different nodes, thus, start of application is considerably slow and not scalable.

Conclusion

- The execution traced using the Extrae, enabled us to discover relevant information about the core operation of *Alya* in the addressed case

- Significant time wasted in the beginning of the execution probably to divide the problem
- Creates overhead since remaining processes are kept idle

- Anomalies were detected in some processes
- Root process (zero) sends the partitioned data sequentially to different nodes, thus, start of application is considerably slow and not scalable.

- Percent imbalance metric indicates that there is a potential performance improvement. The reason for such imbalance is still being investigated.

Future Work

- Study possible change in the code to improve load balance
- Execute Similar experiment with other configurations to confirm results varying:
  - Number of nodes
  - Number of cores
  - Number of timesteps
- Mark the beginning of each iteration so balancing metrics can be applies in each phase of computation/communication
- Trace with different tracing tool that generates smaller trace files e.g. ScoreP

Thank you

Thank you