D5.2.4: Post-processing: revision of system, data format and algorithms definition for exascale systems

WP5: User tools

Project Acronym	CRESTA
Project Title	Collaborative Research Into Exascale Systemware, Tools and Applications
Project Number	287703
Instrument	Collaborative project
Thematic Priority	ICT-2011.9.13 Exascale computing, software and simulation

Due date:	M24
Submission date:	30/09/2013
Project start date:	01/10/2011
Project duration:	36 months
Deliverable lead organisation	DLR
Version:	1.0
Status	Final
Author(s):	Fang Chen (DLR), Derek Groen (UCL)
Reviewer(s)	David Lecomber (ASL), George Mozdzynski (ECMWF), David Henty (UEDIN)

Dissemination level	
PU	*PU – Public*

Version History

Version	Date	Comments, Changes, Status	Authors, contributors, reviewers
0.1	31/08/2013	First version of the deliverable	Fang Chen (DLR)
0.2	17/09/2013	Revision according to review 1.	David Lecomber (ASL)
0.3	18/09/2013	Revision according to review 2	George Mozdzynski (ECMWF)
0.4	23/09/2013	Revision according to review 3	David Henty (UEDIN)
1.0	25.09/2013	Final version submitted	Fang Chen (DLR)

Table of Contents

Index of Figures

Index of Tables

1 Executive Summary

This deliverable reviews the system definition and post processing algorithms that have been proposed in work package 5.2.1. In this work package, we study and evaluate the post-processing system architecture as well as the visualization algorithms. To evaluate the reliability and the compliance of the system, we test the prototyped post-processing tools which were co-designed for HemeLB.

Interactive data exploration and visualization are two major goals in exascale data post-processing. While pre-processing of the simulation focuses on mesh creation and partitioning, post-processing of the simulation is targeted at providing visualisations of the simulation outputs, which serves as a tool to explore and analyse the simulation results.

In an exascale environment, real-time visualisation of the simulation mesh, its partitioning and intermediate simulation results are important for an on-going simulation. It does not only make it possible to analyse intermediate simulation results, but also enables the user to detect and foresee failures in a running simulation process. In work package 5, we focused on developing users tools which provides in-situ and interactive post-processing for analysing running simulations.

In the past months of the ongoing project, work package 5 has established a major collaboration for co-design with **HemeLB,** which is a Lattice-Boltzman based fluid dynamics simulation code. We developed our post-processing system and algorithms in order to provide interactive and in-situ visualization for HemeLB. However, the proposed ideas and algorithms are not limited to this single type of solver. Instead, they can be also applied to other kinds of fluid simulation solvers such as OpenFoam.

2 Introduction

As fluid simulation is heading towards exascale computing, the size and complexity of the simulation data present new challenges to data post-processing frameworks and visualization algorithms. Major challenges in post-processing will be that no data is stored on disk and moving data around will be very costly. Understanding how to make the best use of existing hardware systems and what exascale post-processing algorithms should look like are the questions we are trying to address.

The post-processing of simulation data is a process which transforms simulation output into suitable visual representations. The so called post-processing pipeline typically consists of data extraction, filtering, mapping and visualization stages. Visualization—the creation of vivid pictures from simulation outputs in the form of arrays of numbers—has become an indispensable tool for scientists [1]. At exascale, it becomes a time-consuming process where efficient and interactive visualization can be a challenging task for post-processing.

Interactive exploration and visualisation methods have proven to be successful in analysing large-scale, complex simulation data. In-situ, multi-resolution, level of detail, and region of interests are the keys aspects of doing interactive post-processing. The fundamental structures as well as algorithms provide the user with the possibility to reduce the data to be visualised, to obtain preliminary visualisation results on a coarser mesh, and to further allow result refinement with higher resolution data.

The aim of deliverable 5.2.4 is to review the system requirements, data format and post-processing algorithms proposed in previous deliverables (5.2.1, 5.2.2, 5.2.3). In section 2 and 3, we re-examine the post-processing system architecture and requirements that we identified at the beginning of the project. In section 4, we evaluate the proposed post-processing techniques using the existing HemeLB datasets on various hardware setups. Possible challenges and bottlenecks are identified in section 5. Section 5 also provides details of the work carried out over the last six months, building on the previous deliverables 5.2.3, 5.2.2 and 5.2.1.

3 Reviewing the system definition

In this section, we re-examine the workflow and architecture of our post-processing system. First we discuss the general system of exascale post-processing. Later we will discuss the look of the system for post-processing for HemeLB, (which is our main co-design application).

As mentioned in previous deliverables, we aim to provide a post-processing tool for running simulations that allows the user to interactively explore and analyse the running simulation. In this case, it is not possible for data to be stored on disk. The simulation process continues without outputting results to file.

3.1 Post-processing for exascale in general

A post-processing system might look similar to Figure 1. A visualisation front end is connected to the simulation. This extracts current simulation output, visualises it and then send back signals for further modification. A user is added to the visualisation front-end. The user can interact and navigate through the dataset, and select interesting regions using region of interest boxes. Therefore, only the selected subset of data will be transferred from the simulation to visualisation machine, resulting in a reduced amount of data transfer and time.

A successful post-processing system does not only provide visualizations for the running simulations, it should also allow the user to steer the simulation. The steering of the simulation can be the modification of certain parameters, boundary conditions and mesh qualities.

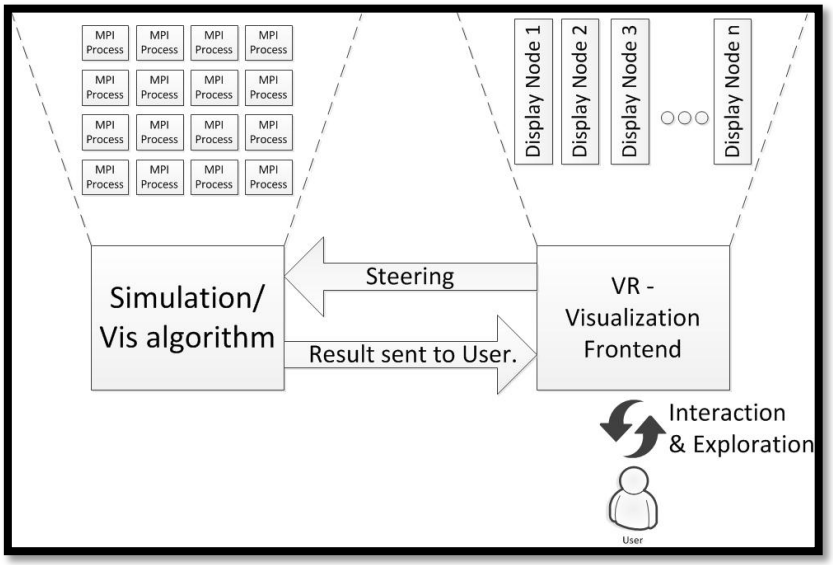

Figure 1 Software architecture. [5]

3.2 Post-processing for HemeLB

The general concept of a post-processing system presented above can be applied to HemeLB in the following way, see Figure 2. In order to make the best use of data distribution, an agreement on the data partitioning has to be reached between pre-processing, post-processing and simulation. In order to avoid re-distribution of data among the nodes, the current post-processing system uses the same data distribution as the simulation itself.

Together with Task 5.1, post-processing for HemeLB provides not only the in situ visualization of a running simulation, but also provides feedback to the pre-processing systems about how the simulation mesh can be refined or further modified.

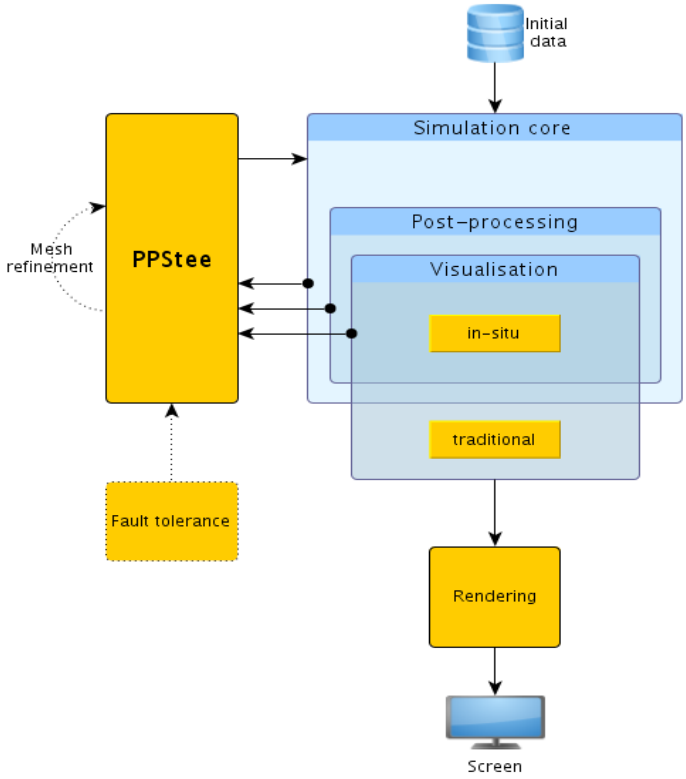

Figure 2 Pre and Post-processing systems together with HemeLB

In the case of a HemeLB simulation, the output of the simulation is not stored at each simulation time step. The output of the HemeLB data provides field information on the lattices. Visualization of the flow therefore must extract the field information at those vertices. A steering client is needed to connect the post-processing tool to the simulation cores.

As the ultimate goal, post-processing combined with pre-processing will provide functionalities such as visualisations and mesh handling. This client or interface should allow the user to analyse the simulation process at run time and further to decide whether and how the simulation meshes can be improved.

4 Reviewing the post-processing algorithms

In this section, we discuss the proposed post-processing algorithms in a general manner. We mainly focus on in-situ and computational steering. Visualisation algorithms on the other hand must be designed to fit the needs of specific simulation applications and is therefore beyond our discussion at this stage.

4.1 In situ visualization

Conventional post-processing is based on simulation outputs that are stored on disk. However, for exascale problems, it is no longer possible to store the complete data on disk and waiting for a simulation to end will take a very long time. Understanding the science behind large-scale simulations requires the extraction of meaningful data from datasets of hundreds of terabytes and more [2]. However, the cost of moving the simulation output to a visualisation machine increases with larger simulations. Therefore, it is preferable to not move the data at all, or to keep the moved data to a minimum [3].

In our proposed system, the simulation and visualization algorithms share the same cluster. At the beginning, the geometry is divided into blocks and distributed among nodes. When the current simulation step is done, a desired visualization is applied on the same block of data, and produces an image data. Then all the images are collected by the master node, and presented in the front end. In this manner, we avoid moving data around and redistributing them among nodes.

4.2 Computational steering

While in-situ and online monitoring provides first insights into the running simulation, computational steering allows further adjustment of the simulation setup. Since exascale systems will add additional requirements on efficiency and power consumption to simulation applications [4], computational steering is a very promising approach to reduce pre- and post-processing efforts by providing the possibility to directly interact and steer a running simulation.

At DLR, we have developed computational steering solutions for HemeLB simulations. A steering client connects the simulation code with visualisation algorithms, rendering systems, and user interaction methods.

The major goal of the steering client for HemeLB is to provide feedback to simulation experts, allowing them to inspect their mesh quality, parameter setups and so on. Together with the pre-processing in Task 5.1, we would like to close a loop among pre-processing, simulation and post-processing. This loop will allow simulation experts to

run their simulation, inspect results at run time, modify simulation configurations and further carry on simulation processes.

One major parameter to steer is the mesh quality. Mesh creation and mesh assessment are not only required to setup a simulation. Many simulations adapt the original meshes to increase the quality of the results of intermediate computation steps. Therefore, concurrent mesh analysis and manipulation methods are additionally needed for interactive online monitoring and computational steering.

In particular, computational steering has to take into account that modifications performed in the visualisation environment are linked to the massively distributed data of a running parallel simulation. Therefore, mapping reordering schemes have to be developed which also work efficiently in heterogeneous exascale environments.

5 Co-design with HemeLB

Following deliverable 5.2.3, we continued the implementation of further visualization techniques which extend the post processing and steering pipeline for HemeLB. Major effort was spent on optimizing the steering client and cut plane prototype. The current steering client is able to access the simulation both at a pausing time step and at a continuous running process, providing an interactive cut plane visualization of the desired simulation steps.

Moreover, having presented the first prototype of cut plane visualizations, we also continued the implementation of further visualization algorithms for HemeLB. One major plan is to provide streamline visualization. The challenges in implementing this algorithm would be the allocation of lattices and the search efficiency for large datasets. This part of the work is still ongoing. A progress update based on the tasks proposed in deliverable 5.2.3 is shown in Table 1.

However, due to unexpected difficulties, we were only able to work with a small testing dataset from the simulation repository. This data is rather a unit-testing dataset which serves as a test to check if the simulation code is running. Such datasets can be used in prototype designs for post processing, for the purpose of learning data formats and data distribution. However, such dataset are far from being sufficient for the demonstration of post processing algorithms.

In order to evaluate the performance of our post processing algorithms, we benchmark the average time needed when additional post processing and steering is connected to HemeLB. These tests have been performed on similar simulation scenarios but with various number of lattices. We will present the results in the following.

Category	Feature, task	Priority	Progress
Steering	Select multiple time steps at a time	Medium	Done
	Switch between single step and non-stop	Medium	Done
	Select a region of lattice	High	95%
	Set a parameter for steering	High	pending
	Send the modified parameter back to Simulation	High	pending
	Connect visualization with steering, define a interface	Medium	100%
Software architecture	Incorporate multi-resolution data structure	Medium	50%

	Try hybrid rendering	Low	pending
	Test remote rendering	Low	pending
User interaction	Allow user to define cutplane location	High	90%
	Combine virtual reality into application. Powerwall, display all.	Low	pending
	Explore the massive data exploration metaphor on powerwall	low	pending
Visualization	Implement stochastic sampling for cut plane points	Medium	30%
	Fix streamline code for HemeLB	Medium	80%
	Compare different image streaming techniques	Low	pending
	Study other suitable visualization tools for HemeLB	Medium	onging

Table 1 Detailed tasks for post processing design and current progress.

The visualization tool has been tested together with the current steering client inside HemeLB on available hardware systems. Six different sizes of regression data sets are tested. These six datasets has similar set ups as shown in Figure 3.

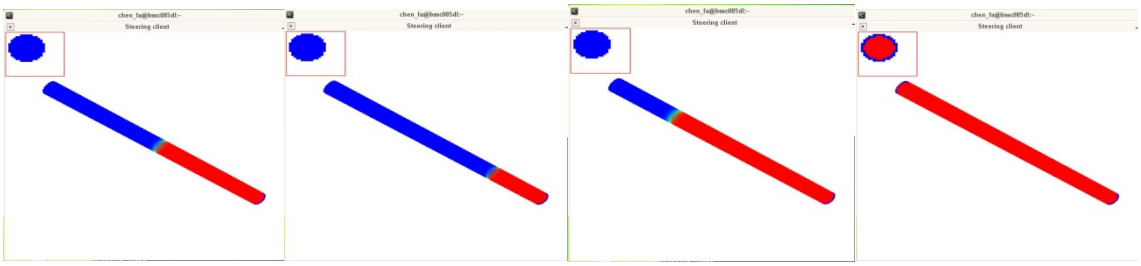

Figure 3 The prototype of cut plane visualization

We measure the computational time that is required per simulation time step, for both a pure simulation run (see Table 2) and with the steering client plugged in (Table 3). This is still ongoing work and we were only able to test our methods on the available data sizes and hardware sizes. The given data sets have between 1 thousand to 2 million lattices.

Number of Lattices	1k	16k	128K	256K	1M	2M

3 proc.	0.0002	0.002	0.019	0.039	0.14	0.20
32 proc	0.00024	0.00103	0.0147	0.0028	0.011	0.02
64 proc	0.00017	0.0014	0.0037	0.0031	0.011	0.02
128 proc	0.00015	0.0019	0.0049	0.0025	0.010	0.02
256 proc	0.0001	0.0021	0.0083	0.0146	0.011	0.021

Table 2 Time measurements (average time needed per time step) for simulation without post-processing connected

Number of Lattices	1k	16k	128K	256K	1M	2M
3 proc.	0.00045	0.0033	0.027	0.05	0.3	0.28
32 proc.	0.00026	0.0012	0.0162	0.0031	0.0153	0.026
64 proc.	0.00018	0.0015	0.0040	0.0028	0.0153	0.027
128 proc.	0.00017	0.00197	0.00493	0.0021	0.0151	0.026
256 proc.	0.00012	0.00219	0.00837	0.0158	0.0149	0.026

Table 3 Time measurements (average time needed per time step) for simulation with post-processing connected

At this stage it is too early to judge the scalability, due to limitations on data sizes and hardware sources. What is important and interesting to us is that adding the steering client and in-situ visualization does not add significant overhead to the computation. We observe a time increase of approximately 20 per cent, which is needed to compute both simulation and the visualization methods. Further work will be carried out to optimize the performance of our post processing algorithms.

Further tests and benchmarking will be carried out as soon as the corresponding large data sets and hardware systems are available. In the following phases, we will also further study and examine the scalability of our post-processing methods.

Currently a simulation with two million lattices is the largest data set available for testing. We expect that communication cost and scalability issues will arise when the data size increases towards an exascale size problem. However, at this stage, we focus on the design of post processing algorithms and visualizations for current available simulations.

6 Conclusion

In this deliverable we have reviewed the proposed system architecture and visualization algorithms which were proposed for exascale post-processing. HemeLB is our main application. We have carried out co-design activities in order to provide interactive and in-situ visualization for running HemeLB simulations. Benchmarks are provided for the currently available simulation test datasets. We present the intermediate results of our in-situ and steering tool for HemeLB. Evidence from the on-going results show that our proposed system is a viable solution for exascale post-processing.

7 References

[1] K.-L. Ma, C. Wang, H. Yu, K. Moreland, J. Huang and R. Ross, “Next-Generation Visualization Technologies: Enabling Discoveries at Extreme Scale,” SciDAC Review, Davis, CA, 2009.

[2] K. L. Ma, R. Ross, J. H. G. Huang, K. Morel and J. Owens, *Utra-scale visualization: Research and education,* 2007.

[3] K. K. Ma, C. Wang, H. Yu and A. Tikhonova, “In-situ processing and visualization for ultrascale simulations,” *Journal of Physics: conference series,* vol. 78, no. 1, pp. 12-43, 2007.

[4] S. Borkar, ““The exascale challenge,” in *International symposium on Vlsi design automation and test (vlsi-dat)*, Taiwan, 2010.

[5] F. Chen, M. Flatken, A. Basermann, A. Gerndt, J. Hetherington, T. Krüger, G. Matura and R. Nash, “Enabling In-situ Pre- and Post-Processing for Exascale Hemodynamic Simulations − A Co-Design Study with the Sparse Geometry Lattice Boltzmann Code HemeLB,,” in *Super Computing*, Salt Lake City, 2012.

[6] T. Tu, H. Yu, L. Ramirez-guzman, J. Bielak, O. Ghattas and K. L. Ma, “From mesh generation to scientific visualization and end-to-end approach to parrallel supercomputing,” in *ACM/IEEE supercomputing conference*, 2006.