SEVENTH FRAMEWORK PROGRAMME
Research Infrastructures

INFRA-2011-2.3.5 – Second Implementation Phase of the European High Performance Computing (HPC) service PRACE

PRACE-2IP

PRACE Second Implementation Phase Project

Grant Agreement Number: RI-283493

D9.3
Emerging Opportunities for Industrial Users of HPC

Final

Version: 1.0
Author(s): Claudio D. Arlandini, CINECA
Date: 22.8.2013

Project and Deliverable Information Sheet

PRACE Project	**Project Ref. №: RI-283493**	
	Project Title: PRACE Second Implementation Phase Project	
	Project Web Site: http://www.prace-project.eu	
	Deliverable ID: < **D9.3**>	
	Deliverable Nature: < Report >	
	Deliverable Level: PU *	**Contractual Date of Delivery:** 31 / August / 2013
		Actual Date of Delivery: 31 / August / 2013
	EC Project Officer: Leonardo Flores Añover	

* - The dissemination level are indicated as follows: **PU** – Public, **PP** – Restricted to other participants (including the Commission Services), **RE** – Restricted to a group specified by the consortium (including the Commission Services). **CO** – Confidential, only for members of the consortium (including the Commission Services).

Document Control Sheet

Document	**Title: Emerging Opportunities for Industrial Users of HPC**	
	ID: D9.3	
	Version: <1.0 >	**Status:** ***Final***
	Available at: http://www.prace-project.eu	
	Software Tool: Microsoft Word 2007	
	File(s): D9.3.docx	
Authorship	**Written by:**	Claudio D. Arlandini, CINECA
	Contributors:	Eva Casoni, BSC; Miroslaw Kupczyk, PSNC; Paride Dagna, CINECA; Ozden Akinci, UHEM; Jan A. Åström, CSC
	Reviewed by:	Dietmar Erwin, JSC; Jussi Heikonen, CSC
	Approved by:	MB/TB

Document Status Sheet

Version	**Date**	**Status**	**Comments**
0.1	17/07/2013	Draft	CA: first draft to partners
0.2	11/08/2013	Draft	CA: general revision by Dietmar Erwin
0.3	12/08/2013	Draft	CA: general revision by Jussi Heikonen
0.4	21/08/2013	Draft	Revision after internal review
1.0	22/08/2013	Final	

Document Keywords

Keywords:	PRACE, HPC, Research Infrastructure, Industry, URANIE, ViscoSolve, NUMFRAC, Ontonix, SPEED, ALYA-Solidz

Disclaimer

This deliverable has been prepared by the responsible Work Package of the Project in accordance with the Consortium Agreement and the Grant Agreement n° RI-283493. It solely reflects the opinion of the parties to such agreements on a collective basis in the context of the Project and to the extent foreseen in such agreements. Please note that even though all participants to the Project are members of PRACE AISBL, this deliverable has not been approved by the Council of PRACE AISBL and therefore does not emanate from it nor should it be considered to reflect PRACE AISBL's individual opinion.

Copyright notices

Table of Contents

Project and Deliverable Information Sheet i

Document Control Sheet i

Document Status Sheet i

Document Keywords ii

Table of Contents iii

List of Figures iv

List of Tables iv

References and Applicable Documents iv

List of Acronyms and Abbreviations v

Executive Summary 1

1 Introduction 2

2 Selection criteria 2

3 ALYA-Solidz 3

3.1 Introduction 3

3.2 Workplan 4

3.3 Implementation 4

3.4 Results 5

3.5 Summary 5

4 NUMFRAC 6

4.1 Introduction 6

4.2 Workplan 6

4.3 Implementation 6

4.4 Results 7

4.5 Summary 7

5 Ontonix 8

5.1 Introduction 8

5.2 Workplan 8

5.3 Implementation 8

5.4 Results 9

5.5 Summary 9

6 SPEED 10

6.1 Introduction 10

6.2 Workplan 10

6.3 Implementation 11

6.4 Results 11

6.5 Summary 11

7 URANIE 12

7.1 Introduction 12

7.2 Workplan 12

7.3 Implementation 13

7.4 Results 13

7.5 Summary 13

8 ViscoSolve ... 14
8.1 Introduction ... 14
8.2 Workplan ... 14
8.3 Implementation ... 14
8.4 Results ... 15
8.5 Summary ... 16
Summary ... 16

List of Figures

Figure 1: ViscoSolve first four iterations total solve time for 127920 hexahedral elements ... 15

List of Tables

Table 1: ViscoSolve parallel performance ... 15

References and Applicable Documents

[1] PRACE-2IP, http://www.prace-project.eu
[2] ALYA, http://www.bsc.es/computer-applications/ALYA-system
[3] SPEED, http://mox.polimi.it/it/progetti/speed/SPEED/Home.html
[4] URANIE, http://sourceforge.net/projects/uranie/
[5] Sáez X., Casoni E., Houzeaux G., Vázquez M., A Parallel Solid Mechanics Solver for multi-physics finite element problems, PRACE-2IP white paper, http://www.prace-project.eu
[6] Åström J. A., Riikilä T. I., Tallinen T., Zwinger T., Benn D., Moore J. C., and Timonen J., A particle based simulation model for glacier dynamics, submitted to “The Cryosphere” (2013)
[7] Zampini, S., Arlandini C., Ontonix: Porting, Optimization and Performance Analysis, PRACE-2IP white paper, http://www.prace-project.eu
[8] Dagna P., Enabling SPEED for near real-time earthquakes simulations, PRACE-2IP white paper, http://www.prace-project.eu
[9] Ostasz, M., Requirements of Industrial Users, D5.1.2 PRACE-1IP Deliverable, http://www.prace-project.eu
[10] METIS, http://glaros.dtc.umn.edu/gkhome/views/metis
[11] Trefethen, L. N., Bau, D., Numerical linear algebra, (1997)
[12] Wall, M. E., Rechtsteiner a., Rocha, L.M., Singular value decomposition and principal component analysis, in „A Practical Approach to Microarray Data Analysis“, pp. 91–109 (2003)
[13] Elemental library, http://libelemental.org
[14] ESSL library, http://www-03.ibm.com/systems/software/essl/
[15] Cornell C.A., Engineering Seismic Risk Analysis. "Bulletin of the Seismological Society of America", 58(6):1583-1606 (1968)
[16] Bouloré A., Struzik, C., and Gaudier F., Uncertainty and sensitivity analysis of the nuclear fuel thermal behavior, “Nuclear Engineering and Design”, 253: 200–210 (2012)
[17] Szpindler, M., Zielinski, M., Applications Addressing Major Socio-economic Challenges, D7.1.1 PRACE-3IP Deliverable, http://www.prace-project.eu
[18] PETSc threads, http://www.mcs.anl.gov/petsc/miscellaneous/petscthreads.html
[19] PETSc libraries, http://acts.nersc.gov/petsc/

List of Acronyms and Abbreviations

BLAS	Basic Linear Algebra Subroutine
BSC	Barcelona Supercomputing Center, Spain
CFD	Computational Fluid Dynamics
CINECA	Consorzio Interuniversitario, the largest Italian computing centre (Italy)
CPU	Central Processing Unit
CSC	Finnish IT Centre for Science (Finland)
DOE	Design Of Experiment
DP	Double Precision, usually 64-bit floating point numbers
EC	European Community
EPCC	Edinburg Parallel Computing Centre (represented in PRACE by EPSRC, United Kingdom)
ESSL	Engineering and Scientific Subroutine Library
FE	Finite Element
GENCI	Grand Equipement National de Calcul Intensif, France
GMPE	Ground Motion Prediction Equation
HLRS	High Performance Computing Centre, Stuttgart
HPC	High Performance Computing; Computing at a high performance level at any given time; often used synonym with Supercomputing
I/O	Input/Output
ITU	Istanbul Technical University, Turkey
JSC	Jülich Supercomputing Centre (FZJ, Germany)
LAPACK	Linear Algebra Package
MKL	Math Kernel Library (Intel)
MPI	Message Passing Interface
OpenMP	Open Multi-Processing
PETSc	Portable, Extensible Toolkit for Scientific Computation
PRACE	Partnership for Advanced Computing in Europe; Project Acronym
PSNC	Poznan Super Computing Centre, Poland
SME	Small Medium Enterprise
SP	Single Precision, usually 32-bit floating point numbers
SVD	Singular Value Decomposition
Tier-0	Denotes the apex of a conceptual pyramid of HPC systems. In this context the Supercomputing Research Infrastructure would host the Tier-0 systems; national or topical HPC centres would constitute Tier-1
UHEM	Turkish National Centre for High Performance Computing

Executive Summary

Work Package 9 (WP9) “Emerging Opportunities for Industrial Users of HPC” of the PRACE-2IP project [1] provided support for industrial applications in order to better exploit HPC resources. Task 9.3 selected applications of high potential impact on industrial research and innovation, but which are not yet ready for industrial production and ported them to suitable systems within the PRACE Research Infrastructure (Tier-1 or Tier-0).

Six codes were identified for enhancement: ALYA-Solidz, a module for the ALYA [2] computational mechanics package aimed at nonlinear continuum structures and large deformation problems; NUMFRAC, a discrete particle method code for the simulation of fractures and granular flows; ONTONIX, a complexity measurement package; SPEED [3], a discontinuous Galerkin spectral element code for seismic wave propagation in complex media; URANIE [4], an uncertainty analysis platform; ViscoSolve, a non-Newtonian flow solver for viscoelastic fluid flow calculations.

The main results are:

- ALYA-Solidz – the solver was parallelized with a MPI/OpenMP hybridisation implementation. A performance improvement of at least 30% was achieved with respect to original MPI code.
- NUMFRAC – a parallel version utilizing domain decomposition and MPI was developed, allowing for 3D simulations.
- ONTONIX – a parallel version of the core Singular Value Decomposition algorithm solver was developed, allowing the handling of systems up to 2. million variables compared to the 10.000 possible with the serial version.
- SPEED – the I/O modules were optimized, and the solver was parallelized with a MPI/OpenMP hybridisation implementation. A linear scalability up to 32.000 threads was obtained, allowing the MunichRE insurance company to apply for a project to a PRACE Tier-0 call.
- URANIE – a robust parallel implementation of the numerical experiments launcher module was designed, as well as a restart procedure for unfinished runs.
- ViscoSolve - the solver was parallelized with a MPI/OpenMP hybridisation implementation.

1 Introduction

Work Package 9 is providing support for applications from industrial partners in order to better exploit HPC resources. Task 9.3 had the particular focus identifying emerging applications, typically open source, of high potential impact on industrial research and innovation, which are not yet ready for industrial production. They were selected through an open process. Input data were collected from both industry and academia to evaluate the promising applications both in terms of industrial interest and HPC challenge. The objective was to port the selected applications to suitable systems within the PRACE Research Infrastructure (Tier-1 or Tier-0), conducting the work in close collaboration with code owners and committed industrial users. The work included performance analysis, scaling, code optimisation, and validation.

The expected outcome is that each of these applications may become the nucleus for a business opportunity for a spin-off or an SME to create an industrial strength offering.

This report discusses the selection process and criteria (Section 2), that resulted in six packages to be considered for the HPC-enabling process. Sections 3 to 8 describe the ALYA-Solidz, NUMFRAC, ONTONIX, SPEED, URANIE and ViscoSolve packages, as well as the outline of the work performed by this task and the main results that were obtained. Finally, Section 9 summarises the work undertaken in years 1 and 2 of task 9.3.

Note that for the six packages, the technical details of the enabling work have been reported in three white papers [5,7,8] already published. Others are in the approval process or submitted to a Journal [6]. So this document will focus mainly on the progress and achievements of the work on those codes. The reader is invited to access the white papers if more in depth technical information is required.

The PRACE partners involved in the technical aspects of the enabling work were: BSC, CINECA, CSC, GENCI, PSNC and UHEM. CINECA coordinated the work in this task.

2 Selection criteria

Strategic guidelines on the type of application best suited for the Task were provided through a preliminary work conducted in collaboration with WP5 of PRACE-1IP, addressing Industrial user requirements and applications used by industry. The joint effort consisted in preparing two surveys, one addressing Industrial End Users and one addressing ISVs. With this approach contacting Industry twice (once for WP5 and once for WP9) to fill in similar surveys has been avoided. The surveys and the results are described in D5.1.2 [11] of PRACE-1IP.

The main insights reached through the surveys pertinent to this Task were that industries strongly felt the lack of appropriate, easy-to-use and inexpensive software, while ISVs were sensible to collaborations to improve the parallel performance of their codes.

Two kinds of parallel applications of interest to industry were therefore identified as main target:

- open source codes already used by industry;
- emerging applications developed in academia with potential interest for industry.

While the survey was an appropriate instrument to understand industrial needs and collect information about applications that have either reached a reasonable maturity level, have been tested by industry, or are in production, it was immediately clear that emerging applications could not be identified with the surveys alone, since they not have reached a sufficient level of maturity or are still in academic incubation. Considering that, the survey could not be the only instrument to identify emerging applications, and the spectrum of contacts needed to be more open than only to industry.

For this reasons all PRACE centres were contacted through a call to promote Preparatory Access projects with relevance to industry. Each centre was requested to conduct enquiries among their academic and industrial contacts. The target of the call was to solicit proposals with the same eligibility criteria as defined in the Preparatory Access call and an accompanying expression of interest from at least one industry partner. Selection criteria addressed the relevance of the application, the coverage of different fields, the level of maturity, and the technical feasibility.

Proposals had to be made by email using a simple template detailing Application Information (proposing institution, name of the application, owner, license model, development status), Project Description (in 200 words maximum), and Expression of interest from industry.

Six proposals were received. A Committee of three WP9 experts was designated by the WP leader for review. The Committee considered the proposals and accepted them all. Some objections were raised regarding Ontonix, since it was not technically compliant to the requirements, being neither an open source code, nor a code developed in academia. The Committee however considered that the requested enabling work addressed only the computational core, which is based on an algorithm which is open-source and widely used in other application domains, as detailed in Section 5. Furthermore, it could be considered a clear example of Open R&D collaboration between PRACE and ISVs to improve the parallel performance of their codes, exactly the need identified by the surveys.

3 ALYA-Solidz

3.1 Introduction

The ALYA System [2], which was developed at the Barcelona Supercomputing Center (BSC-CNS), is a Computation Mechanics (CM) code capable of solving various physics problems, each one with its own modelling characteristics, in a coupled way.

Fluid mechanics simulation codes are generally more advanced in terms of parallel efficiency than solid mechanics codes. One of the reasons is that fluid problems have traditionally required larger meshes than solid mechanics problems. With the increasing need for more advanced modeling techniques involving non-local approaches or multiphysics interactions, Finite Element (FE) solid mechanics codes requirements have traditionally slowed down the parallel efforts aimed at increasing the computational scalability of such codes.

ALYA is organized in modules that solve different physical problems. During the last five years, the main efforts have been dedicated to the fluid mechanics module, covering a wide range of physical problems, such as compressible and incompressible flows, thermal flows, but also excitable media or quantum mechanics for transient molecular dynamics. These modules are now being used in production and scalability has been proven for up to 20.000 CPUs using meshes with billions of elements. The ultimate goal of this work was to improve the solid mechanics module of ALYA to the quality standard already reached by the fluid mechanics modules.

The industrial expression of interest came from Biscarri Consultoria SL, a consultancy and engineering company based in Barcelona, independent from software vendors, devoted to numerical simulations and know-how transfer of CAE technologies to industry and society. They needed a powerful solid mechanics software to give support to customers requiring large simulations.

BSC has led the ALYA-Solidz enabling work.

3.2 Workplan

The performed work focused entirely on the solid mechanics module of ALYA that has been developed to treat large complex problems involving solid deformations. Using the large-scale PDE solver capability of ALYA's kernel, the solid module was implemented so that any future development of more complex FE techniques should conserve the high scalability of the code. These developments have been carried out along two parallel strategies: first, the Message Passing Interface (MPI) implementation, where the parallelization is based on a sub-structuring technique and uses a Master-Slave strategy. Secondly, the OpenMP implementation: in order to handle many-core processors, the OpenMP paradigm has been implemented. The parallel model is therefore a hybrid MPI/OpenMP model.

It is a fact that for implicit methods, the algebraic solver is the key point to obtain a scalable code. If the work is well balanced, the matrix and right hand side (RHS) assemblies scale perfectly. Iterative solvers were used to solve the momentum equation. For symmetric problems (such as the ones studied here), a Conjugate Gradient (CG) method with diagonal preconditioning is used. However, Deflated CG can also be used in order to reduce the number of iterations. Both methods usually perform very well in terms of convergence rate for solving these equations. For non-symmetric matrices, ALYA includes also standard iterative solvers.

Parallelization in ALYA is mainly done via MPI. A domain decomposition strategy has been implemented. It only uses parallelism at task level, which is provided by MPI. To improve efficiency, it was chosen as a strategy for the present enabling work to develop a hybrid code in order to take advantage of all the levels of parallelism that a multicore architecture offers and also to enable one MPI task accessing all the memory of a node. To exploit the thread-level parallelism of multi-core architecture OpenMP has been used in the most time-consuming routines of the solver. Additionally, as the communications between threads are via shared memory, a reduction of the communication cost between processors was expected. Mesh partitioning is automatically done using METIS [12], creating a set of subdomains that communicate via MPI with a typical Master-Slave strategy. On the other hand, internal loops have been parallelized using OpenMP to benefit from memory sharing in multicore clusters.

3.3 Implementation

To exploit the thread-level parallelism of a multicore architecture, OpenMP directives have been added to the MPI parallel code. The first step consists in selecting the most-time consuming routines of the code. Among these routines not all are suitable to be parallelized at the thread-level: they must contain a loop structure with independent iterations. It is important that the loop contains a large code body, since a considerable amount of computational work hides the overhead of thread-management.

In order to introduce thread parallelism in the assembly loop, two main OpenMP directives have been used to tell the compiler how to parallelize it:

- Guided scheduling: since the workload during each iteration is not the same, iterations are assigned to threads in blocks. In a guided scheduling the block size decreases during each iteration (in contrast with the dynamic scheduling), therefore obtaining a better relation between the thread management time and the balanced workload.

- Data scope: the variables that are shared among the iterations are visible to all threads, while the ones that have an independent value among iterations have a private copy per thread.

Typical problems that arise managing thread parallelism are the so-called race conditions, which might lead to nondeterministic results. Basically, race conditions arise when different threads try to update the same state or array in an uncontrolled way. Operations upon shared states are known as critical regions. It is clear that both the operation to assemble an element into a local matrix, and the addition of that local matrix into the global matrix must be thread safe. This data has to be protected by introducing critical regions. Despite the fact that these regions cannot be avoided, it is important to minimize them, because their excessive use can serialize the execution, hence losing the optimal parallel design of the code. The adopted solution is to specify different regions names, combined with the use of atomic clauses for a single memory location.

3.4 Results

In order to demonstrate the performance and the good scalability of the proposed hybrid OpenMP/MPI strategy, a mechanical test involving a large data structure has been used. It consists of a fusion reactor, which is assumed to have a homogeneous isotropic material behaviour.

The computation has been performed on Marenostrum III (BSC), which is a supercomputer based on Intel Sandy Bridge processors that uses a Linux OS. The machine consists of 3028 nodes, two 8-core processors (Intel Xeon E5-2670 cores at 2.6 GHz) per node and 32 GBytes of memory per node. The total Peak Performance is 0.7 PetaFlops. The scalability has been studied as follows: the size of the problem has been kept fixed while the number of cores used in the simulation has been increased (strong scaling). Given the architecture of Marenostrum III, the maximum number of threads per MPI task is 16, since each node consist of two 8-core processors, and the number of MPI tasks will be the number cores divided by 4. The hybrid method shows optimal performance up to 2048 parallel threads but also shows the advantages that a hybrid OpenMP/MPI parallelization provides. See [5] for details.

3.5 Summary

The enabling work addressed the solid mechanics module of ALYA code for solving linear and nonlinear continua and structures problems with large deformations. The main contribution has been the implementation of a new hybrid version of the code with nearly identical behavior of the original ALYA code. To enhance the massively parallel performance OpenMP directives have been added to the current MPI ALYA code. The technique has been tested successfully with a three-dimensional structural test with different solvers, showing optimal scalability results.

4 NUMFRAC

4.1 Introduction

NUMFRAC is a particle based code which has been designed to simulate fracture and fragmentation in nonlinear, visco-elastic and plastic materials. The code has previously been used for modelling, for example, glacier dynamics, brittle fragmentation, cytoskeleton dynamics, and mechanics of fiber networks. A material is modelled as a set of elastic particles that are connected by non-linear interaction potentials. The code allows for arbitrary arranged ’packings’ of particles of different size, kind and shape. For example, a rather loose random packing of spheres of different size and stiffness can be used as a model of an isotropic porous material with structure fluctuations. In contrast, a close packed hexagonal arrangement of identical spheres models a dense uniform anisotropic material. Geometry, interaction, and all other relevant parameters can be set separately for all particles and particle-particle interactions, making the code a tool for investigating mechanical behaviour of strongly disordered materials.

The industrial expression of interest came from Wärtsilä Corporation, a company leader in manufacturing of heavy diesel engines for ships and power plants. They need a tool able to simulate the surface fragmentation in diesel engines known by the name ”fretting”.

CSC has led the NUMFRAC enabling work.

4.2 Workplan

Fretting is a usually harmful process that appears typically between metal parts during frictional wear of surfaces under compressive conditions. Fretting appears under strong enough compression for the frictional sliding to cause fracturing of the metal surfaces. This is typically a problem for engines’ parts. The present enabling work has been devoted to allow NUMFRAC to be used for simulating realistic surface wear in fragile materials like metals by developing a three-dimensional version.

4.3 Implementation

The code does not use any commercial sub-routines or libraries. It is written in Fortran and uses MPI for parallel computing. The scalability to large systems, i.e. many particles, is close to perfect. The origin of this scalability is the short range interaction of the particles combined with a domain decomposition algorithm. The code is optimized using standard techniques from molecular dynamics avoiding N^2 algorithms, where N is the number of particles. A clear majority, about 90%, of the computation time is spent in the force calculations. The most severe limiting factor is the time-step.

There is no useful way in which time can be made parallel and, at present, the code may, in the best case, if we take glacier dynamics as a typical, well known example, simulate about one hour of dynamics in 24 hours of computing. The only reasonable way to speed up the computations is to simulate materials with low viscosity (considerably lower than that of ice).

A typical fretting simulation is expected to take about as long as a glacier dynamics one. In this case the dynamics is very fast and the time steps need to be reduced even further. A typical value is about 10^{-6}. Since computation in time cannot be made parallel, it is important, for computational efficiency to maximize the time-step without violating the stability of the computation. Since there are no explicit temperature fluctuations in the simulations and no connected heat-bath it is easy to check for stability by recording the total energy and check for energy conservation during simulations.

The particle size sets a very intuitive resolution limit for the simulation. For glacier dynamics the particles are of the order of one cubic meter, but the model is inherently scale invariant, therefore nothing much changes if applying to fretting simulations. As long as the equations of motion remain unchanged, all parameters can be re-scaled without changing the simulation results. This means, the same simulation can be view as a e.g. a cubic-meter or cubic-millimeter simulation if all length units are changed accordingly.

In order for the particle model to be able to model not only fracture of elastic material, but also plastic and visco-elastic materials it has been necessary to introduce more elaborate interaction potentials compared to purely elastic ones. The microscopic mechanism behind plasticity and visco-elasticity is that local interactions cannot just be broken but also reform in configurations different from the original one. This is the general principle behind irreversible material deformations like viscotic flow and plastic deformation. In the particle model, plasticity can be introduced via an ordinary yield/fracture stress criterion combined with an additional criterion that allows new contacts between particles to be formed. These new contacts may be formed between any two particles that come close enough to each other as the material deforms. With this mechanism, the material stress will not only be dependent on the load as in an elastic material before any fracture has taken place, but also on the load history.

4.4 Results

Modifying the code to allow 3D simulations was successfully completed. Some unresolved challenges for using NUMFRAC for fretting simulations remain. One is that NUMFRAC is not able at the moment to simulate the very long lasting rapid oscillations that often lead to fretting damage in reality. Simulations are restricted to short times and a few oscillations only. There are also other aspects of fretting, like time dependent viscosity and plasticity which all could be implemented but have not yet been done. For a realistic implementation, a close cooperation with experiments is necessary. The details of the 3D implementation are described in [6].

4.5 Summary

The enabling work addressed the adaptation of the NUMFRAC code for 3D simulations of fractures and fragmentation in nonlinear, visco-elastic and plastic, materials.

While important steps have been completed, further effort is necessary to implement a number of key features in order to allow users to tackle complex problems, like fretting simulations of engines.

5 Ontonix

5.1 Introduction

The Ontonix Software Suite consists of a series of modules: first of all OntoNet, an engine capable of measuring the complexity and other metrics such as robustness and resilience rating of a generic system, where the system is described by N numerical variables. In addition, the Ontonix Software Suite includes other modules with functionalities aimed to support the analysis of the above metrics. The input data processed by OntoNet consist of X tables of data with N columns (variables) and M rows (like observations of historical series). The computing resources needed by OntoNet are very sensitive to the values of X, N and M and when they become large the traditional computers available in the marketplace are not capable to provide the needed computing power.

The industrial expression of interest came from Ontonix SRL, a privately held software and services firm headquartered in Como, Italy. They need a tool able to measure the complexity of a business or a dynamic process of a much larger scale than possible at the moment.

CINECA has undertaken the work in this task.

5.2 Workplan

The aim of the present enabling work has been to adapt the Ontonix Software to a parallel computing environment and allow to compute highly complex problems for very large datasets. This may enable the analysis of complexity of world scale environments/domains, like the complexity of the system made of all enterprises quoted in all stock exchanges around the world.

While some parts of the software are patented, like the pre-processing of input data for the preparation of the input matrix, the core part of the processing algorithm is based on Singular Value Decomposition (SVD) [13] and open-source. Since the SVD method (see [14] for a review) is common to other application contexts, the enabling work and the experience are relevant to other users and communities.

5.3 Implementation

The Ontonix core module requires the computation of the whole eigenvalue spectrum of large real symmetric dense matrices; eigenvectors computation is not needed. Many well-known algorithms exist in literature to solve the problem, and they can be primarily classified as direct or iterative methods. Direct methods perform the task by first reducing the original matrix to tridiagonal form and then computing the eigenvalues of the resulting tridiagonal matrix. On the other hand, iterative methods progressively compute the eigenvalue spectrum by means of matrix-vector multiplications and Gram-Schmidt orthogonalizations and are mainly designed for problems where the matrix is sparse and the number of required eigenvectors is modest; therefore, direct methods should be preferred in this case. Moreover, it thus becomes clear that the total available RAM and the total RAM per core are the computational bottlenecks of the problem. In the spirit of not reinventing the wheel, the development of an ad-hoc code from scratch to perform the task was avoided being very time consuming and prone to numerical stability issues.

Within direct methods, different approaches can be taken into account. Since there is no software on the market able to perform an out-of-core computation of the eigenvalue spectrum with a distributed memory computation, the parallelization work has been based on libraries for distributed memory High Performance Linear Algebra. The library Elemental [15] was chosen as the most efficient, for both the reduction to tridiagonal form, and the solution of the tridiagonal eigenvalue problem.

5.4 Results

The experimental results were obtained on CINECA BlueGene/Q machine FERMI (10.240 power A2 sixteen-core nodes running at 1.6 GHz and equipped with 16 GB of RAM). Elemental has been compiled using XL compilers, with optimization level -O3 in a pure MPI mode without pthread or OpenMP support, using the ESSL library [16] as BLAS implementation and a full LAPACK implementation (i.e. not the one provided by the ESSL library).

Since the computational bottleneck of the problem of interest is the available memory, a test case was used to find the largest dimension of the local matrices. In detail, the driver has been run with a 4x4 grid of MPI processes restricting the computation on a single FERMI node. The largest matrix size possible turned out to be 40.000 rows, which means that the greatest allowed size for each MPI process is 10.000 rows (equivalent to a memory occupation of 750 MB in double precision).

A weak scalability test has then been performed to investigate the parallel efficiency of the algorithm implemented by Elemental; more precisely, the local problem size was kept fixed (10.000 rows) and the parallel driver has been run with a different number of MPI processes, ranging from 16 to 1'024. An extrapolation was then obtained up to 32.768 MPI processes by fitting the computational times collected from the weak scalability test with a second-order polynomial in the number of MPI processes. This approach suggests that a complexity analysis with the Ontonix code will be solvable using the CINECA FERMI BlueGene/Q machine if the matrix size is below 2.560.000 rows.

For details see [7].

5.5 Summary

In this work the possibility of solving the real symmetric dense eigenvalue problem required by the Ontonix project was investigated using a distributed memory approach on the BlueGene/Q machine FERMI. The state-of-the-art High Performance library for Dense Linear Algebra Elemental has been used to fulfil the task. Preliminary results suggest that analysis of highly complex systems are already doable with Tier-0 systems. To go further, other techniques should be investigated if both of the following requirements can be fulfilled: a) the full eigenvalue spectrum is not required and b) the matrix can be sparsified. In such a case, iterative methods can be successfully used together with an out-of-core implementation of the matrix-vector product.

6 SPEED

6.1 Introduction

SPEED (Spectral Element in Elastodynamics with Discontinuous Galerkin) is an open source code, jointly developed by the Department of Structural Engineering and of Mathematics at Politecnico di Milano, for seismic hazard analyses.

The variety and extent of impacts caused by the destructive earthquakes of recent years on the built and natural environment revealed dramatically the need for improving the tools for seismic risk assessment. In the last twenty years there has been impressive progress worldwide towards the development of deterministic ground shaking scenarios as input within seismic hazard and risk assessment studies, relying on numerical simulation of seismic wave propagation under realistic tectonic and geo-morphological conditions.

Nowadays, standard seismic hazard analysis is achieved through a well-consolidated approach, namely Probabilistic Seismic Hazard Analysis [17]. As a basic ingredient, the method requires a suitable tool for predicting the earthquake ground motion produced at the target site by any potential earthquakes along any possible rupture areas of the seismogenic source. The standard approach is through Ground Motion Prediction Equation (GMPE). However, despite their simplicity, GMPEs may not be suitable to reproduce specific ground motion features (e.g., near fault and complex site effects), due to the scarcity of the calibration data set in the near field of an earthquake or for very large subduction events.

Therefore, under such conditions, the only way to estimate accurately the seismic hazard might be to adopt a more physical approach capable to limit the systematic bias between data and prediction. However, to include in a single physics-based model the coupled effects of the seismic source, the propagation path through complex geological structures and localized superficial irregularities, such as alluvial basins or synthetic infrastructures, still poses challenging demands on computational methods and resources due to the coexistence of very different spatial scales, from a few tens of kilometers, with reference to the seismic fault, down to a few meters, or even less, when considering some structural elements. The SPEED code was developed to deal with these challenges.

The industrial expression of interest came from both MunichRE, one of the largest re-insurance companies of the world, based in Munich, who need modern tools for risk analysis, and MOXOFF, a Politecnico di Milano spinoff, who need a powerful software to give support to customers requiring earthquake engineering simulations.

CINECA has led the SPEED enabling work.

6.2 Workplan

SPEED is a discontinuous Galerkin spectral element code that incorporates the open-source libraries METIS and MPI for the parallel computation (mesh partitioning and message passing).

Before starting the present enabling work, SPEED allowed only regional or local simulations, a typical mesh covering an area of 50x50x30 km^3, with a run time of 10 to 20 hours on 64 parallel threads. The optimization of the code for use on a petaflop system was therefore planned to allow:

- analysing larger geographical areas (e.g.: the Maule earthquake in Chile covering a volume of 1000x500x800 km^3)

- evaluating the impact on complex geological structures (like strata few tens meters wide) or large engineering structures (bridges, buildings, …)
- parametrical studies on relevant physical quantities, like earthquake magnitude, rift typology, rift-breaking typology, …

The code works on both structured and unstructured meshes. A run is made of two different phases, a serial one for pre-processing of data and mesh preparation, followed by a computational parallel one. Benchmarks executed on Linux clusters showed that the computational phase maintained an acceptable scalability up to 512 threads. The first phase was very significant and especially for unstructured meshes might amount to the majority of the total run-time.

The enabling work concentrated then on two aspects:

- optimizing the pre-processing phase;
- improving scalability by modifying the parallel section from completely MPI to a hybrid MPI/OpenMP architecture.

6.3 Implementation

A thorough profiling of the code was made, identifying the main routines necessary of a deeper investigation. Looking carefully at the logic of the algorithm it was found out that some loops and conditional clauses were not necessary. Loops were therefore rearranged, in order to make operations made on different elements independent. Thus, it was possible to add, at this level, a further Open-MP parallelization. An optimized hybrid version of SPEED was then obtained by creating a parallel region before the loop over subdomain elements.

The optimized version not only has allowed the improvement of SPEED performance in terms of the overall computational time, but has also solved a great memory constraint present in the pure MPI version. Because of the huge amount of memory per MPI process required to simulate real earthquake scenarios, only a subset of the available cores per node could be used, wasting a lot of computing resources. With the new hybrid solution, MPI tasks and OpenMP threads can be mixed to fully exploit the computing power of the IBM PowerA2 processors.

6.4 Results

The test case considered for final benchmarking was the 29th May 2012 magnitude 6.0 earthquake that struck the Po Plain, Emilia-Romagna, north-eastern Italy. The model extends over a volume of about 74x51x20 km^3 and is discretized using an unstructured hexahedra mesh with characteristic element size ranging from ~ 150 m at the surface to ~ 1400 m at the bottom of the model.

With respect to the original pure-MPI version, an overall 5x speed-up of the key routines was obtained. Furthermore, while the original version presented acceptable scalability up to 1,000 threads, the new version was tested up to 16,000 threads maintaining almost linear speed-up.

For details see [8]

6.5 Summary

The enabling work produced a new hybrid version of SPEED with high parallel efficiency, giving the possibility to run simultaneously parametric simulations on a Tier-0 machine, dramatically reducing the execution time and making such deterministic simulations feasible, producing reliable "real-time" results.

The enabling work allowed MunichRE to submit a large project proposal to the PRACE 7th Tier-0 call, immediately showing the industrial relevance. The proposal was recently awarded 40.000.000 core hours on the FERMI CINECA system. The results of 3D ground-motion simulations obtained with SPEED will be used to improve the physical reliability of the numerical scenarios, allowing, on one side, to estimate more precisely the seismic hazard (i.e.: the spatial correlation of ground motion), on the other side, to understand which physical parameters (i.e.: directivity, slip pattern) play a crucial role in terms of potential damages to the infrastructures.

7 URANIE

7.1 Introduction

Over the last decade, the improvements of computer hardware and software have brought a significant change in the capabilities of simulation software in the field of nuclear applications. New computer power made possible the emergence of simulations that are more realistic (complex 3D geometries being treated instead of 2D ones), more complex (multi-physics and multi-scale being taken into account), and more meaningful (with propagation of uncertainties).

In order to treat uncertainty analysis in this constantly evolving framework, CEA has developed a software platform named URANIE [4,18] that provides tools for validation, optimization, uncertainty analysis, model calibration for high performance computing codes.

The industrial expression of interest came from CEA itself, requiring a version able to deal with larger datasets.
GENCI has led the URANIE enabling work, with the collaboration and technical work done by PSNC.

7.2 Workplan

URANIE is based on the realisation of DOEs (Design Of Experiments), i.e. a set of simulations in which the same code is executed with slight modifications in the input files, so that the uncertainty range of the input variables is covered.

In order to accommodate various middlewares and launch codes as black boxes, the URANIE launcher has the following strategy:

- a single job is allocated,
- the master node runs a control process,
- the control process launches children of the control process, each child running an mpirun script which runs one computation in the DOE,
- the control process checks for the state of the children in order to decide when to run new children.

This strategy gives good flexibility and good performance on hundreds of processors, but the bottleneck on the master node can become a problem for large runs. Also, this strategy gives little control on the placement of processes in the context of coupled simulations.

The present enabling work aimed at improving URANIE on two issues:

- Capacity to launch efficiently large DOEs
- Capacity to provide checkpoint/restart mechanisms for large DOEs that were incompletely run.

7.3 Implementation

The enabling work on the URANIE software has been performed on PSNC GPGPU cluster CANE.

A careful analysis of the software architecture was performed, especially directed at these key libraries:

- DataServer library, which defines the TDataServer object which contains all the information about the uncertain variables for the analyses;
- Sampler library, which is devoted to generate a design of experiment (deterministic/statistical) from characteristics of the uncertain variables. Several methodologies are implemented:
- qMC ("quasi Monte-Carlo") sequences (Sobol, Halton);
- SRS (“Simple Random Sampling”), LHS (“Latin Hypercube Sampling”), ROA ("Random Orthogonal Array"), Archimedean Copulas;
- MCMC ("Markov Chain Monte-Carlo") method for Gaussian mixture.
- Launcher library, which is devoted to manage the computation on a desktop (sequential) or on a cluster (distributed). The goal is to construct the Y matrix jointed in the X matrix of the design of experiment;
- Modeler library, which is devoted to build a surrogate model from input to output attributes contained in a database;
- Optimizer library, which is devoted to perform a Verification and Validation code or to find the optimum of a computation code or analytical function

7.4 Results

Although significant steps were made, problems related to the installation of URANIE on the CANE cluster, did not allow completing all the expected software optimizations. Work is still progressing in the framework of PRACE-3IP WP7 (Applications Addressing Major Socio-economic Challenges) [19]. URANIE was chosen due to the code application to safety analysis of nuclear power plants.

7.5 Summary

URANIE provides tools for validation, optimization, uncertainty analysis, model calibration for high performance computing codes. The work undertaken in the this task aimed to make the software evolve so that it will be suitable for exploitation of design of experiments implying thousands of cores in multiple contexts: serial codes, parallel codes, coupled simulations.

8 ViscoSolve

8.1 Introduction

ViscoSolve is a stable unstructured finite volume method code for parallel large-scale simulation of viscoelastic fluid flows, developed at the Istanbul Technical University (ITU). The numerical method is based on the side-centered finite volume method where the velocity vector components are defined at the mid-point of each cell face, while the pressure term and the extra stress tensor are defined at element centroids. The main advantage of ViscoSolve approach is that it leads to a stable numerical algorithm and does not require any ad-hoc stabilization techniques. Therefore, the continuity equation is satisfied at the machine precision.

The industrial expression of interest came from Arcelik, the major Turkish producer of household appliances, interested in tools for design and modelling of polymer products.

ITU-UHEM has led the ViscoSolve enabling work.

8.2 Workplan

ViscoSolve has the potential to accurately predict the onset of viscoelastic fluid flow instabilities which is very important for many industrial polymer processes where output quality constraints require that operating conditions should be maintained in the stable flow regime. Therefore, the tool has the potential for increasing the polymer processing speed while reducing the cost. In addition, it should help to reduce a huge industrial polymer processing wastage due to product defects which are caused by viscoelastic fluid flow instabilities.

The code has been written based on MPI. Preconditioned iterative solvers are based on the PETSc library for improving the efficiency of the parallel code.

The present enabling work aimed to optimize the code performance, allowing the simulation of more complex test cases. The challenge was dealt with by adapting the code to a hybrid MPI/OpenMP architecture.

8.3 Implementation

The code has three main steps. The first one is a grid subroutine that reads mesh files and specifies the required boundary conditions. The second step consists of the setup subroutine that constructs the algebraic matrices for the Stokes system and sets up the geometric non-nested multigrid solver for the Stokes system. These first and second steps are called only once at startup. The third step is the numerical time integration part which dominates the solver calculation time. A typical time dependent calculation requires approximately 1,000 to 10,000 iterations to obtain the final steady-state or the time-periodic state.

Note that the implementation of the preconditioned iterative solvers is based on the PETSc library [17] for improving the efficiency of the parallel code. PETSc is based on 3.0.0 version which is not thread safe [18], therefore thread based parallelization has not been implemented on PETSc routines

A careful scrutiny of all steps has been performed in order to establish the best implementation of OpenMP directives.

8.4 Results

The enabling work was performed on the PRACE Tier-1 system Karadeniz at UHEM. The Karadeniz system has 64 compute nodes, each offering two 4 core Intel Xeon 5550 (Nehalem) processors with a nominal clock speed of 2.67 GHz. Each node of Karadeniz has 24 GB RAM, with InfiniBand (20Gbps) connection.

The parallel performance of the ViscoSolve algorithm was tested using the unstructured computational mesh with 1.279.200 hexahedral elements for the viscoelastic fluid flow past a confined circular cylinder.

The parallel performance of the ViscoSolve algorithm is presented in Table 1: ViscoSolve parallel performance for the initial setup phase and the total solve time for the first 4 iterations. The first column represents the number of threads, the second column shows the initial setup time for the construction of the matrices for the Stokes system and the time required for the construction of the two-level non-nested multigrid method, the third column shows the total solve time for the first four iterations (one number for each iteration), the fourth column shows the number of multigrid iterations for the Stokes system and the fifth column represents the number of iterations for the restricted additive Schwarz preconditioner to solve the extra stress tensor. The final sixth column “N” represents the number of subdomain ratio between fine and course mesh levels. Total times of the first four iterations are shown in Figure 1.

A good scalability is therefore reachable up to 256 parallel threads.

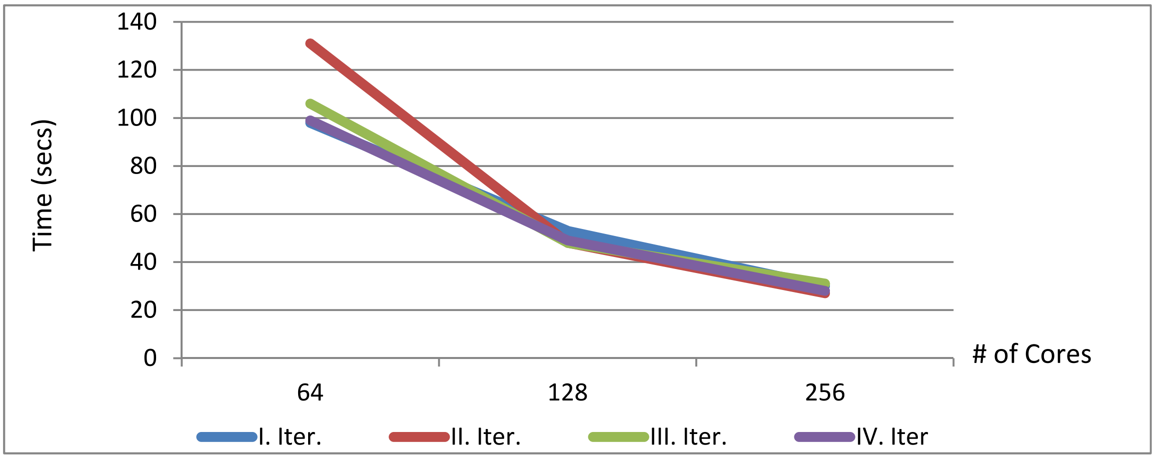

Figure 1: ViscoSolve first four iterations total solve time for 127920 hexahedral elements

# of threads	Setup time	Total solve time (secs)	MG iteration #	ASM ILU(0) iteration #	N
64	27	98/131/106/99	100/72/86/77	0/8/8/8	2
128	14	53/48/48/49	100/74/75/77	0/9/9/9	4
256	8	30/27/31/28	100/76/78/77	0/9/9/9	8

Table 1: ViscoSolve parallel performance

Implementing a hybrid strategy was more challenging than thought at the beginning. The work presents interesting results, but a more careful implementation needs to be completed in order to obtain the performance required for real industrial test cases.

8.5 Summary

The preliminary results confirm that ViscoSolve is a powerful tool for predicting the onset of viscoelastic fluid flow instabilities, which is very important for many industrial polymer processing.

Summary

In the first year the activity was dominated by the gathering of information about industrial needs via the surveys and the establishment of the enabling projects.

The second year focused on the implementation phase. The main results in this second year are:

- ALYA-Solidz – solver was parallelized with a MPI/OpenMP hybridisation implementation. A performance improvement of at least 30% was achieved with respect to original MPI code.
- NUMFRAC – a parallel version utilizing domain decomposition and MPI was developed, allowing for 3D case simulations.
- ONTONIX – a parallel version of the core Singular Value Decomposition algorithm solver was developed, allowing the handling of systems up to 2.500.000 variables, compared to the 10.000 possible with the serial version.
- SPEED – I/O modules were optimized, and solver was parallelized with a MPI/OpenMP hybridisation implementation. A linear scalability up to 32.000 threads was obtained, allowing MunichRE insurance company to present a project to a PRACE Tier-0 call, recently awarded with 40.000.000 core hours on the PRACE Tier-0 CINECA system.
- URANIE – a robust parallel implementation of the numerical experiments launcher module was designed, as well as a restart procedure for unfinished runs.
- ViscoSolve – solver was parallelized with a MPI/OpenMP hybridisation implementation

Although PRACE-2IP is coming to an end, in all cases the work on these codes is planned to continue. In the case of URANIE, this will be done in the framework of PRACE-3IP WP7 (Applications Addressing Major Socio-economic Challenges) [16], due to the code application to safety analysis of nuclear power plants. MunichRE is funding a postdoc grant to the Politecnico di Milano university to expand functionalities of the SPEED code. In the other cases, the industrial partners are still evaluating the results providing test cases of industrial relevance to the involved computing centres. The in depth knowledge gained working with the codes on HPC systems will therefore likely continue to be applied, for the on-going benefit of both industry and academic users. With respect to the expected outcome, that each of these applications may become the nucleus for a business opportunity to create an industrial strength offering, while some tangible results have already been obtained (like MunichRE being awarded 40.000.000 core hours on PRACE Tier-0 systems), an impact analysis will be possible only on a time scale well beyond the end of the PRACE-2IP project.