ISSN 1392 – 124X INFORMATION TECHNOLOGY AND CONTROL, 2010, Vol. 39, No. 4

SOLUTION AND ANALYSIS OF CFD APPLICATIONS BY USING GRID INFRASTRUCTURE

Arnas Kačeniauskas

Laboratory of Parallel Computing, Vilnius Gediminas Technical University
Saulėtekio St. 11, Vilnius, LT-10223, Lithuania
e-mail: arnas.kaceniauskas@vgtu.lt

Abstract. The paper describes solution and analysis of complex computational fluid dynamics (CFD) applications by using grid infrastructure. Powerful and flexible computing environment including computing resources, data storage, simulation software, visualization e-service and graphical job monitoring tool is developed and employed for solution of CFD problems. Efficiency of parallel computations is evaluated by performing a benchmark based on convective pollution transport. Challenging dam break flow simulation including highly non-linear breaking waves requires all advanced features of modern BalticGrid infrastructure. The numerical results are validated by quantitative comparison with the experimental measurements.

1. Introduction

Computational fluid dynamics (CFD) plays an important role in the design of modern industrial components [2]. Many industrial applications described by the Navier-Stokes equations are solved by CFD codes [27]. Numerous examples include actual problems such as oil filters, pollution transport, sluice gates and tank sloshing. Various computational methods are developed for discretizing the governing equations and spatial domains. The finite element method (FEM) [30] has the capability of accepting complex geometries in an integrated fashion, making it particularly interesting to the designers of complex mechanisms. However, it has always been the problem of the FEM that larger computational times have been associated with it. Grid computing [6] is thus perceived as a promising avenue for future advances in this applied area of science.

European Grid Infrastructure [4] can provide resources for solving large industrial CFD problems. Grid computing represents one of the most promising advancements for modern computational science. With the power of grid, scientists are able to perform simulations at previously impossible and unexplored problem scales. Grid development efforts push paradigms of remote HPC resource usage [7]. Scientists who used to login to the supercomputer of their choice to submit computing jobs are now presented with an interface that can assign their computational workload to a pool of resources anywhere in their Virtual Organization (VO). However, this leads to very complicated environments handling complex simulation on remote heterogeneous architectures [21]. The design of distributed parallel algorithms and the software deployment in grid [5] presents a new challenge to computational scientists. Recent progress resulted in the dramatic increase of computational capacity, which approached the petaflop level.

Analysis of complex CFD problems requires powerful, universal and flexible computing environment including computing resources, data storage, simulation software, job submission and monitoring tools, graphical user interfaces like web portals and visualization system [12, 21]. The desktop-delivered visualization and grid computing might become the solutions to provide sufficient performance, visualizing a relatively large dataset with the help of relatively cheap hardware. The accomplishment of this task by using ordinary personal computers reveals a challenge, but is highly appreciated in academic communities. Visualization systems have become an essential part of the emerging fabric of grid services. However, remote grid visualization leads to very complicated software systems. For example, RealityGrid project [25] selected VTK [26] as a lower-level environment, along with enabling technologies such as Chromium [14]. Most of grid environments for visualization [1, 19] are based on the Globus middleware [10] and its toolkit for service development. Visualization software can be highly integrated with working environment.

BalticGrid [3] infrastructure is built on gLite middleware [9], developed within EGEE. Only part of Globus functionality can be accessed in the considered grid environment, therefore, most of available visualization software, web portals and graphical user interfaces cannot be directly applied.

The paper presents powerful and universal computing environment employed for solution and analysis of CFD applications. Software integration and deployment issues are covered in details. Grid visualization e-service VizLitG is developed for interactive visualization of remote data files located in grid storage elements. The performance analysis reveals how efficiently CFD applications can be solved on gLite based grid infrastructure.

2. Description of investigated CFD problems

Several actual CFD applications like oil filters, sediment transport, oil reservoirs and dam break flows have been investigated on BalticGrid infrastructure. In this section, two pilot CFD applications are described. A rotating cone problem originated from the investigation of convective pollution transport can be considered as a test problem for stabilisation methods and parallel algorithms. A dam break problem includes complex breaking wave phenomena that require advanced modelling techniques, impressive computing resources and flexible simulation software.

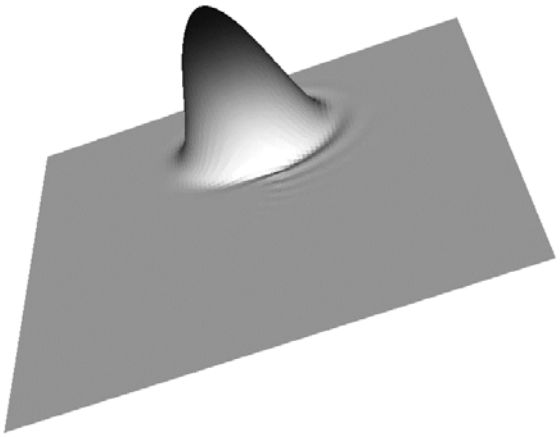

Figure 1. Rotating cone after 2π period of time

The rotating cone problem (Figure 1) is widely used to illustrate the effectiveness of algorithms in case of convection dominated flows. The 2D square solution domain [-0.5; 0.5]x[-0.5; 0.5] is discretized by structured finite element meshes. The concentration cone of radius 0.15 is positioned at (-0.25; 0.0). In the centre of the cone, the concentration maximum of 1.0 decreases to zero as sinusoidal curve. The velocity field $u = -y$, $v = x$ corresponds to a rotational flow with a nature of a solid body. The problem is numerically difficult to solve not only because of the pure advection but also because of the numerical diffusion attributed to the Cartesian grid discretizing the rotational flow field.

The dam break problem including the breaking wave phenomena has been the subject of extensive research for a long time [18, 22, 24]. However, the universal, accurate and efficient numerical technique for breaking wave simulation attracts big attention of research community and software developers. Measurements of the exact interface shape are not available, but some secondary data such as reduction of the water column height can be employed for quantitative comparison of the results [22].

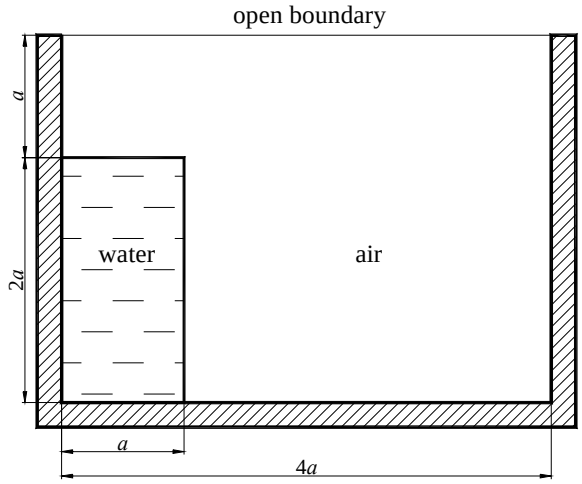

Figure 2. Geometry of the dam break problem

The geometry of the solution domain is shown in Figure 2. The dimensions of the reservoir and the water column correspond to those used in the experiment carried out by Koshizuka *et. al*. [18]. The reservoir is made of glass, with a base length of 0.584 m. The water column, with a base length of 0.146m and the height of 0.292 m (a = 0.146 m), was initially supported on the right by a vertical plate drawn up rapidly at time t = 0.0 s. The water falls by gravity (g = 9.81 m/s^2), acting vertically downwards. The density of water is ρ_A = 1000 kg/m^3, while the dynamic viscosity coefficient is μ_A = 0.01 kg/(m · s). The density of air is taken to be ρ_B = 1 kg/m^3, and the dynamic viscosity coefficient is μ_B = 0.0001 kg/(m · s).

3. Governing equations

The laminar and Newtonian flow of viscous and incompressible fluids is considered. It is governed by the Navier-Stokes equations (the Eulerian reference frame)

$$\rho\left(\frac{\partial u_i}{\partial t}+u_j\frac{\partial u_i}{\partial x_j}\right)=\rho F_i+\frac{\partial \sigma_{ij}}{\partial x_j}, \quad (1)$$

$$\frac{\partial u_i}{\partial x_i}=0, \quad (2)$$

where u_i are the velocity components; ρ is the density; F_i are the gravity force components and σ_{ij} is stress tensor

$$\sigma_{ij} = -p\delta_{ij} + \mu\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right), \quad (3)$$

where μ is dynamic viscosity coefficient; p is pressure and δ_{ij} is Kronecker delta.

The pseudo-concentration method [28] is developed for moving interface flows using the Eulerian approach and the interface capturing idea. The pseudo-concentration function φ serves as a marker, identifying fluids A and B with densities ρ_A and ρ_B and viscosities μ_A and μ_B. In this context, the density and viscosity are defined as:

$$\rho = \varphi\rho_A + (1-\varphi)\rho_B, \quad (4)$$

$$\mu = \varphi\mu_A + (1-\varphi)\mu_B, \quad (5)$$

while $\varphi = 1$ *for* fluid *A* and $\varphi = 0$ for fluid *B*.

The evolution of the moving interface as that of pollution concentration is governed by a time dependent convection equation

$$\frac{\partial\varphi}{\partial t} + u_j\frac{\partial\varphi}{\partial x_j} = 0. \quad (6)$$

The initial conditions defined on the entire solution domain should be prescribed for the equation (6). In case of equations (1)-(3), the slip boundary conditions are prescribed on rigid walls while the zero stress boundary conditions are prescribed on the upper open boundary. The detailed discussion on stress dependent boundary conditions and their implementation can be found in the work [16].

The space-time Galerkin least squares finite element method [23] is applied as a general-purpose computational approach to solve the partial differential equations (1)–(6). The detailed description of variational formulation and stabilisation parameters can be found in the work [17].

4. Software deployment on grid infrastructure

The discussed applications have been solved by the code FEMTOOL [17], designed for coupled problems of CFD. FEMTOOL allows implementation of any partial differential equation with minor expenses. Time dependent problems are solved using space-time finite elements. The order of shape functions is determined by input and is limited neither in space nor in time. A given transient problem can be solved in several implicit time steps from one time level to the other or in one single implicit step for all time levels. Space-time finite element integration in time and the high order shape functions generated automatically make FEMTOOL to be applicable to complex CFD applications of interest.

Several times FEMTOOL software was deployed on gLite based grid. The gLite Workload Management System (WMS) natively supports the submission of MPI jobs, which are jobs composed of a number of processes running on different Working Nodes (WN) in the same Computing Element (CE). However, this support is still experimental, therefore, running MPI applications on the gLite grid requires significant hand-tuning for each site. The first FEMTOOL deployment relied on shell scripts consisting of approximately 100 lines. Only experienced system administrator can prepare such scripts for new MPI applications. Recent efforts made in Interactive European Grid project slightly improved the situation [15].

The latest FEMTOOL gridification is based on the compilers g95 and gcc as well as on the OpenMPI implementation. FEMTOOL version 3.0 has been deployed in BalticGrid-II testbed by using SGM (Software Grid Manager) system. All software packages are installed in the predefined location, which is specified by content of *$VO_BGTUT_SW_DIR* variable. After successful installation, the SGM marks the site in global grid information system as capable of running FEMTOOL application. By use of the flag *VO-balticgrid-A-ENG-FEMTOOL-3.0*, called "a tag" the ordinary FEMTOOL users may indicate which sites they want to use.

Solution of large and complex CFD applications on heterogeneous grid infrastructure requires graphical user interface that hides the complexity of the grid middleware from the ordinary user and makes access to the grid resources easy and transparent. Migrating Desktop Platform [20] developed at the Poznan Supercomputing and Networking Centre has been applied to run CFD applications on BalticGrid infrastructure. The Migrating Desktop Platform is a powerful and flexible user interface to grid resources that gives a transparent user work environment and easy access to resources and network file systems independently of the system version and hardware. It allows the user to run applications, manage data files, and store personal settings independently of the location or the terminal type. User can perform basic operations on files (downloading, uploading, transferring or removing, etc.) stored on data management systems like gLite SEs and GridFTP servers.

The key feature of the Migrating Desktop Platform is the possibility of easy adding various tools, applications and different visualization formats. The Migrating Desktop offers a framework that can be easily extended on the basis of a set of well-defined plug-ins used for: accessing data, defining job parameters and pre-processing job parameters. Special plug-in has been developed in order to integrate FEMTOOL within Migrating Desktop Platform. The same plug-in covers rotating cone problem and dam break problem. A user selects the problem, enters its governing parameters, defines suitable grid resources and submits job to grid infrastructure by using Job Submission Wizard (Figure 3). The plug-in writes problem parameters to

the metadata file of FEMTOOL, prepares the file defining the job, creates shell script file specifying executable with arguments and submits job to grid. Wizards of Migrating Desktop Platform handle the whole job's life-cycle from job defining and submission to job state monitoring and visualization of job results by provided software. Finally, computed results are transferred to the most suitable Storage Element.

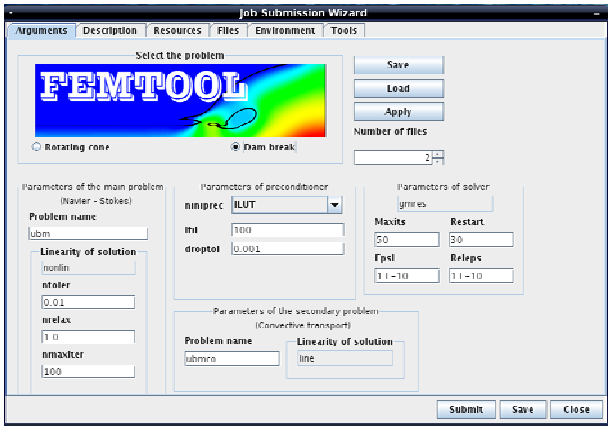

Figure 3. FEMTOOL plug-in of Migrating Desktop

5. Visualization of remote result files

The obtained results have been processed by grid visualization e-service VizLitG designed for convenient access and interactive visualization of remote results located in Storage Elements. VizLitG (Figure 4) is developed and maintained in the Laboratory of Parallel Computing of Vilnius Gediminas Technical University.

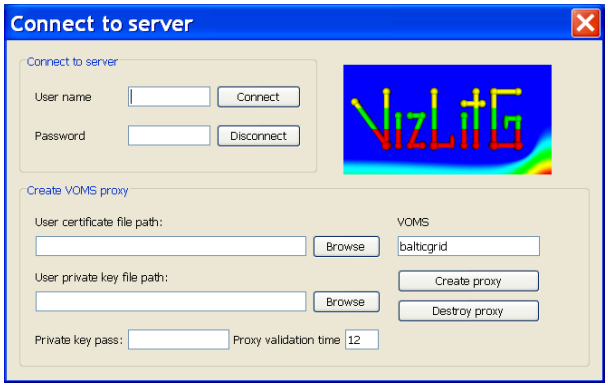

Figure 4. Connection dialog of grid e-service VizLitG

The client-server architecture of the grid visualization service is based on widely recognized web standards. VizLitG has flexible environment and remote instrumentation of e-service provided by Java and GlassFish application server [8]. The visualization engine of the VizLitG is based on VTK toolkit [26]. VTK modules are enwrapped by Java programming language and build the service running on the server. The visualization server runs on the special User Interface named UIG (User Interface for Graphics). User authentification and full data transfer from SE is performed by gLite means. Moreover, remote instrumentation of e-service provides for users flexible access of remote data files located in grid. Transfer of interactively selected parts of datasets located in experimental SE instead of the whole data files can save significant amount of visualization time.

The developed GUI allows automatic data management and interactive dataset selection. In order to process HDF5 [13] files automatically, datasets are stored in predefined structure allowing the software to interpret the structure and contents of a file without any outside information. HDF5 groups and datasets are automatically processed considering values of HDF5 attributes. Several API's for writing data files in predefined HDF5 format are provided for VizLitG users. C and FORTRAN90 programming languages are considered as well as C++. Time dependent and time independent data are processed differently. Datasets that do not change in time are located in the group Common. Datasets varying in time are grouped and stored according to time step number. GUI separates geometry and topology from attributes like scalars or vectors in order to emphasize their different nature (Figure 5).

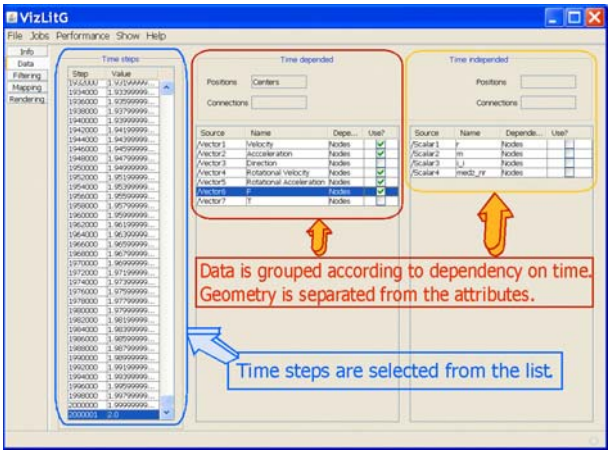

Figure 5. The data window for interactive dataset selection

A visualization network is assembled from VTK objects by using GUI. The resulting pipelines are described by XML language [29]. Valid XML documents are automatically generated on a client and transferred to the server by JAX-WS (Java API for XML Web Services) Runtime. Tabular design of GUI fields simply illustrates the dataflow. Multiple VTK renderers can work in one render window. VizLitG allows grouping of complex visualizations in several viewports. The user can assign considered mappers to different viewports.

In order to provide for remote users high interactivity level, GVid software [11] was implemented in VizLitG as video-streaming module. The most important GVid classes were renewed to support VTK 5.6 and enwrapped by Java. The compressed video stream is efficiently transferred through the network and displayed on the client. Implemented VTK widgets

(Figure 6) provide for remote users of the e-service full interactivity.

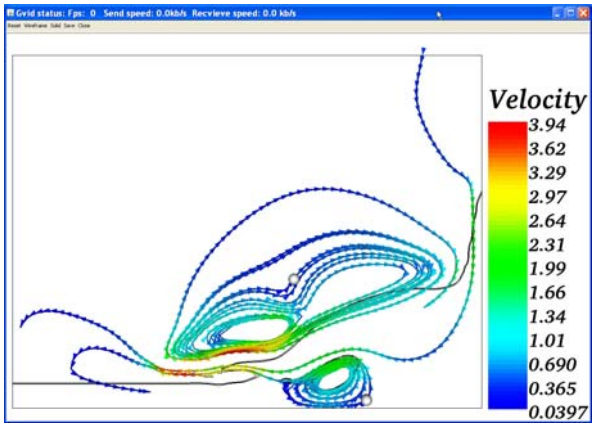

Figure 6. Interactive widget applied to plot stream lines

6. Numerical results and discussions

Computations have been performed on the different sites of BalticGrid infrastructure [3]. These sites provide for BalticGrid users over 5300 CPU and 120 TB of data storages. Some of the employed Computing Elements belong to Lithuanian NGI (National Grid Initiative) tightly incorporated into European Grid Infrastructure.

The parallel performance of computations has been evaluated by measuring the speed-up S_p

$$S_p = \frac{t_1}{t_p}, \qquad (7)$$

where t_1 is the program execution time for a single processor; t_p is the wall clock time for a given job to execute on p processors. Parallel speed-up has been measured by fixing the number of particles and increasing the number of the processors used.

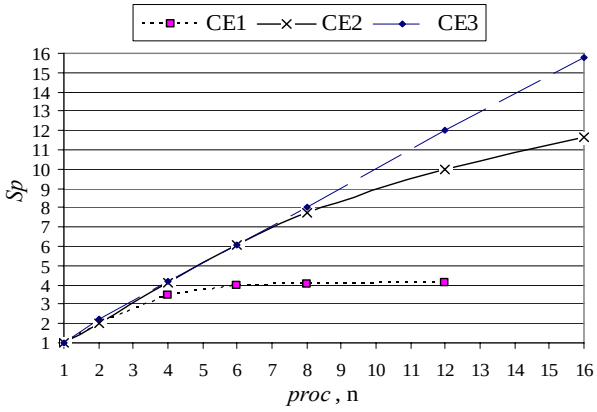

Figure 7. Spead-up measured on different BalticGrid CEs

The considered benchmark is based on numerical solution of the rotating cone problem discretized by 160000 finite elements. The results of speed-up analysis obtained on three CEs are presented in Figure 7. The speed-up (7) as a function of the number of processors is plotted. Advantages obtained by parallelism on the site CE1 diminish rapidly as the number of processors exceeds some threshold value. This behaviour is explained by the fact that the parallel speed-up is largely determined by the ratio of local computations over inter-processor communication. As the number of processors increases, for a fixed problem size, the communication cost will eventually become dominant over the local computation cost after a certain stage. It is evident that the discussed threshold value for site CE1 is unacceptably small. The obtained results can be explained by the fact that the site CE1 is collected from the old hardware including slow 100Mbit/s network.

Speed-up measured on other sites CE2 and CE3 is significantly higher. When the number of processors is small, the measured speed-up is close to linear on both sites. The reduction of the efficiency owing to communication overhead is obtained for a larger number of processors on CE2. Nearly ideal speed-up is observed on CE3 site for the considered number of processors, which is caused by perfectly balanced ratio of communications to computations.

The dam break problem is also investigated by performing numerical experiments on sustainable BalticGrid infrastructure. The computations are performed by using the structured finite element meshes of different resolution – 120×90 and 240×180. The investigated time interval is t = [0.0; 1.0]s. The size of the time step is Δt = 0.001667 for the 120×90 finite element mesh. The number of time steps is equal to 600. The size of the time step for the 240×180 finite element mesh is Δt = 0.000667. The number of the time steps used is equal to 1500. The large number of numerical parameters governing the sharpness of the front and mass conservation should be investigated in order to obtain accurate numerical solution. Available grid infrastructure is well suited for required parametric computations.

Gravity causes the water column on the left of the reservoir to seek the lowest possible level of potential energy. Thus, the column will collapse and eventually come to rest. The initial stages of the flow are dominated by inertia forces. On such a large scale, the effect of surface tension forces is insignificant. The complexity of velocity fields, occurring at different stages of breaking wave phenomena, can be easily captured using simple structured meshes.

The numerical results have been validated by the quantitative comparison with experimental measurements obtained for the early stages of dam break problem [22]. Non-dimensional position of the leading edge of the collapsing water column on the left wall versus non-dimensional time $t^* = t\sqrt{2g/a}$ is shown in Figure 8. A thin layer of water shoots over the bottom and the rest of the bulk flow follows behind it. The accurate numerical results (the curve Num) have been obtained by applying the interface sharpening technique.

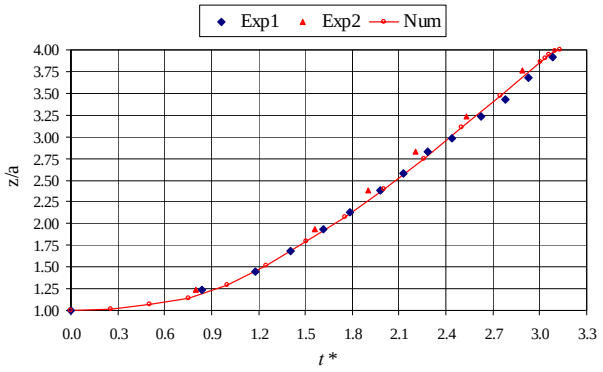

Figure 8. Quantitative comparison of the numerical results and experimental measurements

Figure 9 shows the screenshot of grid visualization e-service VizLitG illustrating the breaking wave phenomenon often occurring in later stages of the dam break problem. Computed velocity field is represented by glyphs. Moving interface is captured by the iso-countour. Grey colours illustrate the pressure field while the chart plot shows variation of the pressure at the bottom. When t=0.83s, the backward moving wave folds over twice and small amounts of air are trapped. However, in experiments, this air is present in the form of small bubbles. The developed methodology has been derived for sharp interfaces, therefore, the mesh needs significant refinement to a resolution smaller than the bubble size.

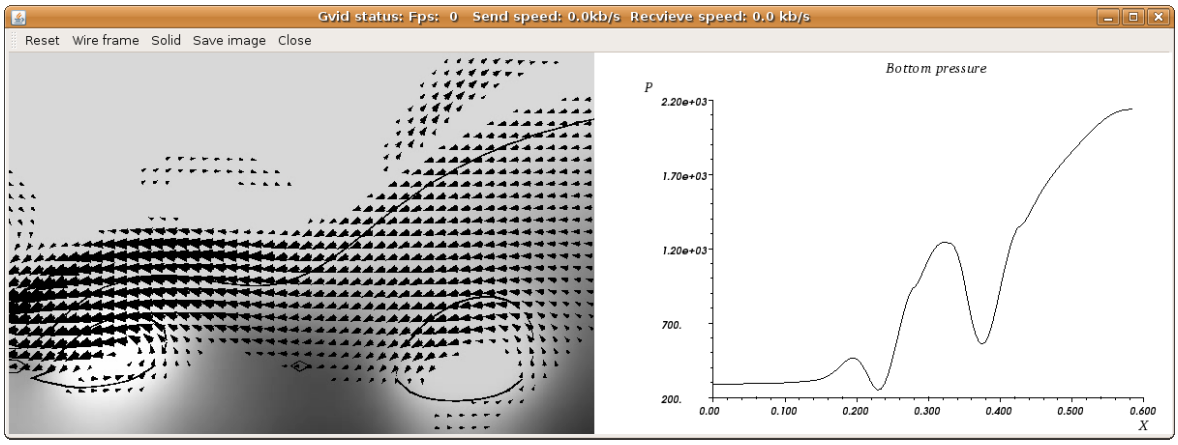

Figure 9. Visualization of breaking wave phenomena

6. Conclusions

In this paper, analysis of complex CFD problems by using grid infrastructure has been described. Development and application issues of modern computing environment have been discovered emphasizing software deployment on grid, automatic job submission and visualization of remote result files. Performance tests based on parallel solution of rotating cone problem revealed that grid sites for efficient computations should be considered carefully evaluating potential abilities of employed hardware. The implemented domain decomposition strategy is well designed for parallel solution of convective transport problems. Challenging dam break flow simulation including highly non-linear breaking waves has required all advanced features of modern BalticGrid infrastructure. The numerical results have been validated by quantitative comparison with the experimental measurements. The computed position of the leading edge of the collapsing water column has been in good agreement with the experimental data. Performed investigation proves that the developed BalticGrid infrastructure provides powerful and flexible computing environment for fast and efficient solution of challenging CFD applications.

Acknowledgement

The work described in this paper is supported by the European Union through the FP7- INFRA-2007-1.2.3: e-Science Grid infrastructures contract No 223807, project "Baltic Grid Second Phase (BalticGrid-II)".

References

[1] **A.A. Ahmed, M.S.A. Latiff, K.A. Bakar, Z.A. Rajion**. Visualization Pipeline for Medical Datasets on Grid Computing Environment. *Proc. of* 5*th Int. Conf. on Computational Science and Applications, IEEE Computer Society Press*, 2007, 567–575.

[2] **A.J. Baker.** Finite Element Computational Fluid Mechanics. *McGraw-Hill*, 1983.

[3] **BalticGrid:** *http://www.balticgrid.eu/* (*Accessed June* 2010).

[4] **EGI**: *http://web.eu-egi.eu/* (*Accessed June* 2010).

[5] **L. Field, E. Laure, M.W. Schulz.** Grid deployment experiences: grid interoperation. *J. Grid Computing*, 2009, *Vol*. 7(3), 287–296.

[6] **I. Foster, C. Kesselman.** Grid: Blueprint for a new computing infrastructure (1st ed.). *San Francisco, Morgan Kaufmann publishers*, 1998.

[7] **W. Gentzsch.** Grid and cloud portals for design, simulation and collaboration. I*n Parallel, distributed and grid Computing for engineering* (*Eds. Topping, B.H.V., Ivanyj, P.*), Saxe-Coburg Publications, 2009, 83–116.

[8] **GlassFish:** *https://glassfish.dev.java.net/* (*Accessed April* 2010).

[9] **gLite:** *http://glite.web.cern.ch/glite/* (*Accessed June* 2010).

[10] **Globus:** *http://www.globus.org/* (*Accessed June* 2010).

[11] **GVid:** *http://www.gup.jku.at/gvid/* (*Accessed July* 2010).

[12] **C.D. Hansen, C.R. Johnson.** The Visualization Handbook. *Elsevier*, 2005.

[13] **HDF5:** *http://hdf.ncsa.uiuc.edu/products/hdf5/* (*Accessed July* 2010).

[14] **G. Humphreys, M. Houston, R. Ng, R. Frank, S. Ahern, P.D. Kirchner, J.T. Klosowski**. Chromium: A Stream Processing Framework for Interactive Rendering on Clusters. *ACM Transactions on Graphics*, 2002, *Vol*. 21(3), 693–702.

[15] **int.eu.grid:** *http://www.interactive-grid.eu/* (*Accessed April* 2010).

[16] **A. Kačeniauskas, R. Kutas**. Implementation of stress dependent boundary conditions in FEM code for coupled problems. *Information Technology and Control*, 2008, *Vol*. 37(1), 69–74.

[17] **A. Kačeniauskas, P. Rutschmann.** Parallel FEM software for CFD problems. *Informatica*, 2004, *Vol*. 15(3), 363–378.

[18] **S. Koshizuka, H. Tamako, Y. Oka.** A particle method for incompressible viscous flow with fluid fragmentation. *Computational Fluid Dynamics*, 1995, *Vol*. 4(1), 29-46.

[19] **D. Kranzlmuller, G. Kurka, P. Heinzlreiter, J. Volkert.** Optimizations in the Grid Visualization Kernel. *Proc. of the Workshop on Parallel and Distributed Computing in Image Processing, Video Processing and Multimedia, IPDPS* 2002, *Ft. Lauderdale, Florida*, 2002.

[20] **M. Kupczyk, R. Lichwała, N. Meyer, B. Palak, M. Plóciennik, P. Wolniewicz.** "Applications on demand" as the exploitation of the Migrating Desktop. *Future Generation Computer Systems*, *Vol*. 21(1), 2005, 37–44.

[21] **M. Li, M. Baker.** The Grid: Core Technologies. *Wiley*, 2005.

[22] **J.C. Martin, W.J. Moyce.** An experimental study of the collapse of liquid columns on a rigid horizontal plane. *Philosophical Transactions of the Royal Society of London*, 1952, *Vol*. A244, 312–324.

[23] **A. Masud, T.J.R. Hughes.** A Space – Time Galerkin/Least – Squares Finite Element Formulation of The Navier-Stokes Equations for Moving Domain Problems. *Computer Methods in Applied Mechanics and Engineering*, 1997, *Vol*. 146(1–2), 91–126.

[24] **M. Quecedo, M. Pastor, M.I. Herreros, J.A. Fernández Merodo, Q. Zhang.** Comparison of two mathematical models for solving the dam break problem using the FEM method. *Computer Methods in Applied Mechanics and Engineering*, 2005, *Vol*. 194(36-38), 3984-4005.

[25] **RealityGrid**: *http://www.realitygrid.org/* (*Accessed June* 2010).

[26] **W. Schroeder, K. Martin, B. Lorensen.** Visualization Toolkit: An Object-Oriented Approach to 3D Graphics, 4th Edition. *Kitware. Inc.*, 2006.

[27] **T.E. Tezduyar.** Finite Elements in Fluids: Special Methods and Enhanced Solution Techniques. *Computers & Fluids*, 2007, *Vol*. 36(2), 207–223.

[28] **E. Thompson.** Use of Pseudo-Concentrations to Follow Creeping Viscous Flows During Transient Analysis. *International Journal for Numerical Methods in Fluids*, 1986, *Vol*. 6(10), 749–761.

[29] **XML:** *http://www.w3.org/XML/* (*Accessed June* 2010).

[30] **O.C. Zienkiewicz, R.L. Taylor.** The Finite Element Method. *Vol*. 1–3, *the fifth edition, London, Butterworth Heinemann*, 2000.

Received August 2010.

DOI: 10.5755/j01.itc.39.4.12383