XXVII CILAMCE

Proceedings of the XXVII Iberian Latin American Congress on Computational Methods in Engineering
September 3 to 6, 2006 - Belém, Pará - BRAZIL.

PARALLEL STABILIZED FINITE ELEMENT SIMULATION OF FREE SURFACE FLOWS WITH VIOLENT MOTIONS

Renato N. Elias
Alvaro L. G. A. Coutinho
Marcos A. D. Martins
{renato, alvaro, marcos}@nacad.ufrj.br
Center for Parallel Computations and Department of Civil Engineering
Federal University of Rio de Janeiro, P. O. Box 68506,
RJ 21945-970 – Rio de Janeiro, Brazil

Abstract. *Flows with violent free-surface motions occur in several problems in hydrodynamics, such as fuel or water sloshing in tanks, waves breaking in ships, offshore platforms, harbors and coastal areas. The computation of such highly nonlinear flows is challenging since free-surfaces commonly present merging, fragmentation and breaking parts, leading to the use of interface capturing Eulerian approaches. In such methods the surface between two fluids is captured by the use of a marker function which is transported in a flow field. In this work we present a 3D parallel incompressible SUPG/PSPG finite element method to cope with free-surface problems within the interface-capturing context. The incompressible flow and transport problems are solved in a segregated manner. Turbulence effects are modeled with a classical Smagorinsky model. We introduce a Parallel Dynamic Deactivation algorithm to solve the marking equation only in a small region around the interface. The implementation is targeted to distributed memory systems with cache-based processors. The performance of the proposed solution method was tested with the classical Dam Break problem.*

Keywords. Free-surface, interface-capturing, finite element, volume-of-fluid

1 INTRODUCTION

Flows with violent free-surface motions occur in many hydrodynamics problems. Sloshing in fuel on water tanks, wave breaking in ships, offshore platforms, harbors, coastal areas, green water on decks are important examples of such problems [17]. The computation of such highly nonlinear free-surface flows is difficult because the position and shape of the free-surface is not known *a priori*. Besides, the free-surface may present merging, cusps and fragmentation processes which make the Lagrangian methods based on mesh adaptation unfeasible [18, 19]. In this work we extend our state-of-art parallel three-dimensional unstructured grid incompressible flow solver to cope with such flows, by implementing an interface-capturing method, where two fluids (e. g. water and air) are considered a single effective fluid with varying properties. The interface is captured as a region of sudden change in the fluid properties.

The main characteristics of our incompressible flow solver [13, 15, 16] are: semi-discrete stabilized finite element formulation; implicit time marching scheme with adaptive time stepping control; advanced Inexact Newton solvers; edge-based data structures to save memory and improve performance; support to message passing and shared memory parallel programming models; large eddy simulation extensions using a simple Smagorinsky model. Into this flow solver we introduce Volume-of-Fluid (VOF) extensions to track the evolving free-surface [4, 19]. The advection equation for the free-surface marker function is solved by an edge-based semi-discrete SUPG finite element formulation with shock-capturing. The computational effort to solve the marking function is limited to a narrow band around the free-surface by a dynamic-deactivation (DD) scheme [6].

The remainder of this paper is organized as follows: the first and second sections present the incompressible flow and interface-capturing governing equations respectively; the third section summarizes the solution procedures employed and the fourth shows the results obtained with the proposed scheme. We simulate the dam-break problem, where a water column is suddenly released. We compare our results with available experimental and numerical results obtained with different methods (finite differences, finite volumes, finite elements, smooth particle hydrodynamics) showing that the present scheme is fast, simple and accurate. The final remarks and conclusions are summarized in the last section.

2 INCOMPRESSIBLE FLOW GOVERNING EQUATIONS

Let $\Omega \subset \mathbb{R}^{n_{sd}}$ be the spatial domain, where n_{sd} is the number of space dimensions. Let Γ denote the boundary of Ω. We consider the following velocity-pressure formulation of the Navier-Stokes equations governing the incompressible flow of two immiscible fluids:

$$\rho\left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u} - \mathbf{f}\right) - \nabla\cdot\sigma = \mathbf{0} \qquad \text{on } \Omega \tag{1}$$

$$\nabla\cdot\mathbf{u} = 0 \qquad \text{on } \Omega \tag{2}$$

where ρ and $\mathbf{u}$ are the density and velocity, $\mathbf{f}$ is the body force vector carrying the gravity acceleration effect and $\boldsymbol{\sigma}$ is the stress tensor given as

$$\sigma(p,\mathbf{u}) = -p\mathbf{I} + \mathbf{T} \tag{3}$$

where p is the hydrostatic pressure, $\mathbf{I}$ is the identity tensor and $\mathbf{T}$ is the deviatoric stress tensor

$$\mathbf{T} = 2\bar{\mu}\varepsilon(\mathbf{u}) \tag{4}$$

and $\varepsilon(\mathbf{u})$ is the strain rate tensor defined as

$$\varepsilon(\mathbf{u}) = \frac{1}{2}\left(\nabla\mathbf{u} + (\nabla\mathbf{u})^T\right) \tag{5}$$

In the present work, the turbulence was modeled as a Large Eddy Simulation (LES) method by the use of a simple Smagorinsky turbulence model. In this model, the viscosity μ is augmented by an eddy viscosity μ_{Smag} which is defined as

$$\mu_{\mathrm{Smag}} = \left(C_S\bar{\Delta}\right)^2 \left|\varepsilon(\mathbf{u})\right| \qquad (6)$$

where C_S is a constant and $\bar{\Delta}$ is the element length defined here as the cubic root of the element volume. In this work C_S was set to 0.1.

The essential and natural boundary conditions associated with equations (1) and (2) can be imposed at different portions of the boundary Γ and represented by,

$$\mathbf{u} = \mathbf{g} \qquad \text{on } \Gamma_g \qquad (7)$$

$$\mathbf{n}\cdot\sigma = \mathbf{h} \qquad \text{on } \Gamma_h \qquad (8)$$

where Γ_g and Γ_h are complementary subsets of Γ.

Let us assume following Tezduyar [11] that we have some suitably defined finite-dimensional trial solution and test function spaces for velocity and pressure, $S^h_{\mathbf{u}}$, $V^h_{\mathbf{u}}$, S^h_p and $V^h_p = S^h_p$. The finite element formulation of equations (1) and (2) using SUPG and PSPG stabilizations for incompressible fluid flows can be written as follows: Find $\mathbf{u}^h \in S^h_{\mathbf{u}}$ and $p^h \in S^h_p$ such that $\forall \mathbf{w}^h \in V^h_{\mathbf{u}}$ and $\forall q^h \in V^h_p$:

$$\int_\Omega \mathbf{w}^h \cdot \rho\left(\frac{\partial \mathbf{u}^h}{\partial t} + \mathbf{u}^h \cdot \nabla\mathbf{u}^h - \mathbf{f}\right) d\Omega + \int_\Omega \varepsilon\left(\mathbf{w}^h\right):\sigma\left(p^h, \mathbf{u}^h\right) d\Omega - \int_\Gamma \mathbf{w}^h \cdot \mathbf{h}\, d\Gamma + \int_\Omega q^h \nabla\cdot\mathbf{u}^h d\Omega$$
$$+\sum_{e=1}^{n_{el}} \int_{\Omega^e} \left[\tau_{SUPG}\rho\mathbf{u}^h \cdot \nabla\mathbf{w}^h + \tau_{PSPG}\nabla q^h\right]\cdot\left[\rho\left(\frac{\partial \mathbf{u}^h}{\partial t} + \mathbf{u}^h\cdot\nabla\mathbf{u}^h\right) - \nabla\cdot\sigma\left(p^h, \mathbf{u}^h\right) - \rho\mathbf{f}\right] d\Omega^e$$
$$+\sum_{e=1}^{n_{el}} \int_{\Omega^e} \tau_{LSIC}\nabla\cdot\mathbf{w}^h \rho\nabla\cdot\mathbf{u}^h \Omega^e = 0$$

(9)

In the above equation the first four integrals on the left hand side represent terms that appear in the Galerkin formulation of the problem (1)-(8), while the remaining integral expressions represent the additional terms which arise in the stabilized finite element formulation. Note that the stabilization terms are evaluated as the sum of element-wise integral expressions, where n_{el} is the number of elements in the mesh. The first summation corresponds to the SUPG (Streamline Upwind Petrov/Galerkin) term and the second to the PSPG (Pressure Stabilization Petrov/Galerkin) term. We have evaluated the SUPG and PSPG stabilization parameters according to Tezduyar *et al.* [12], as follows:

$$\tau_{SUPG} = \tau_{PSPG} = \left[\left(\frac{2\left\|\mathbf{u}^h\right\|}{h^{\#}} \right)^2 + 9 \left(\frac{4\nu}{\left(h^{\#}\right)^2} \right)^2 \right]^{-1/2} \tag{10}$$

Here $\mathbf{u}^h$ is the local velocity vector, ν represent the kinematic viscosity and the "element length" $h^{\#}$ is defined to be equal to the diameter of the sphere which is volume-equivalent to the element.

In Equation (9), the last summation is the Least Squares Incompressibility Constraint (LSIC) term [10], added to prevent oscillations in high Reynolds number flows. The LSIC stabilization parameter, following [10] is

$$\tau_{LSIC} = \frac{\left\|\mathbf{u}^h\right\| h^{\#}}{2} \tag{11}$$

The discretization of Equation (9) leads us to a nonlinear system of equations to be solved at each time step.

3 INTERFACE-CAPTURING GOVERNING EQUATIONS

In the volume-of-fluid method, also called pseudo-concentration method, the main idea is to define a scalar marking function $\phi(\mathbf{x},t)$ over the computational domain in such a manner that its value at a certain point $\mathbf{x} \in \Omega$ and instant $t \in [0,t_f]$ indicates the presence or absence of fluid. Thus, let us assume the value 1 to regions filled with a fluid A, e.g., water, and the value 0 to regions filled with a fluid B, e.g., air. The position of the fluid interface will be defined by the isovalue contour $\phi(\mathbf{x},t)=\phi_c$, where $\phi_c \in [0,1]$ is its critical value defined a *priori*. The value $\phi_c=0.5$ is usually assumed. Therefore, the function $\phi(\mathbf{x})$ driven by a velocity field $\mathbf{u}$ can be computed with the following transport equation in conservative form:

$$\frac{\partial \phi}{\partial t} + \nabla \cdot \left(\mathbf{u}\phi\right) = 0 \tag{12}$$

In the volume-of-fluid (VOF) formulation the fluid density and viscosity, employed in the fluid flow solution, are interpolated across the interface as follows:

$$\rho = \phi(\mathbf{x},t)\rho_1 + \left[1-\phi(\mathbf{x},t)\right]\rho_0 \tag{13}$$

$$\mu = \phi(\mathbf{x},t)\mu_1 + \left[1-\phi(\mathbf{x},t)\right]\mu_0 \tag{14}$$

where subscripts 0 and 1 denote the values corresponding to each phase.

The finite element formulation of Equation (12) can be written as follows: Find $\phi^h \in S_\phi^h$, such that, $\forall w^h \in V_\phi^h$:

$$\int_{\Omega} w^h \left(\frac{\partial \phi^h}{\partial t} + \mathbf{u}^h \cdot \nabla \phi^h \right) d\Omega + \int_{\Omega} w^h \left(\nabla \cdot \mathbf{u}^h \right) \phi^h d\Omega$$

$$+ \sum_{e=1}^{n_{el}} \int_{\Omega^e} \tau_{SUPG} \mathbf{u}^h \cdot \nabla w^h \left(\frac{\partial \phi^h}{\partial t} + \mathbf{u}^h \cdot \nabla \phi^h + \left(\nabla \cdot \mathbf{u}^h \right) \phi^h \right) d\Omega^e \tag{15}$$

$$+ \sum_{e=1}^{n_{el}} \int_{\Omega^e} \delta \nabla w^h \cdot \nabla \phi^h d\Omega = 0$$

where the first two integrals represent the Galerkin formulation of Equation (12), while the first element-wise summation represent the SUPG (Streamline Upwind Petrov-Galerkin stabilization) and the second summation term is due the CAU (Consistent Approximated Upwind) shock capturing term to reduce the interface smearing during its advection. The evaluation of τ_{SUPG} and δ stabilization terms followed the definitions described in [9] and [8] respectively. The discretization of Equation (15) leads us to a non-linear, due the shock capturing term, ordinary differential equation system.

4 SOLUTION PROCEDURE

The computational solution kernels consist of a predictor/multicorrector time integration scheme as described in [9, 14] for both, incompressible fluid flow and interface transport equations. The nonlinearities due to the convective term on the Navier-Stokes equation and the shock capturing on the transport equations, are treated by an Inexact Newton-GMRES scheme as described in Elias *et al*. [15]. Moreover, since both physics are discretized in an implicit manner, the use of a Proportional-Integral-Derivative (PID) controller based on feedback controlling theory allows the time step to be chosen according to the solution error (see [5] for further details). Most of the computational effort spent during the solution phase is due to the matrix-vector products within the GMRES driver. In this sense, we adopt an edge-based data structure in order to minimize indirect memory addressing, diminish the floating point operation counts (flops) and memory requirements as described in Elias [15, 13]. Further computational gains are obtained from data preprocessing performed by the EdgePack library – a package to optimize data based on reordering and grouping techniques [20]. All computational kernels previously cited are hybrid parallelized covering most of the High Performance Computing techniques such as distributed and shared memory systems as well as vectorized and pipelined processors [16].

We introduce here another computational artifact to further improve the overall efficiency of the present free-surface solver, the parallel dynamic deactivation (PDD) technique for solving the marking function. This technique is an extension of the dynamic deactivation (DD) procedure, which is an algorithm that restricts the computation on regions were high gradients are present. It was firstly presented by Lohner in [6] for contaminant transport problems and is based on the same partitioned operator idea behind adaptive implicit/explicit methods such as that described by Souza *et al* in [7]. In this paper we have extended these works to deal with parallel free surface flows. Since the marking function employed on volume-of-fluid methods presents steep gradients, the dynamic deactivation algorithm catches and restricts the computations only on regions around the interface. Therefore, most of the computational effort that would be necessary to solve the interface transport over the whole domain is considerably saved. Moreover, a buffer zone around the interface is built to assure that the interface is kept within the enabled region in each time step. It is important to emphasize that although the computational costs associated to the transport problem are recognizably lesser than those spent by the Navier-Stokes solution, a similar approach can be employed to restrict the computations only on regions filled by the aimed fluid during the incompressible solution phase. The set of active elements initially selected by the DD algorithm is based upon the following criteria:

$$\|\nabla\phi\|^e \geq \|\nabla\phi\|_m \tag{16}$$

where $\|\nabla\phi\|^e$ is the Euclidean norm of the element gradient solution while $\|\nabla\phi\|_m$ is the average norm gradient computed for the whole computational grid. After the initial selection, a buffer zone is built by selecting a number of layers, n_{layers}, with active elements. In this work, all computations were carried out with n_{layers} set to 5. Note that this parameter can be set according to the CFL condition in order to assure that interface does not cross the buffer zone. In the PDD extension, each processor estimates its own number of finite element entities (elements, edges and nodes) enabled for computations. This is illustrated in Figure 1, where we show the evolution of the active elements in the solution of the pure advection of a circle. If there are no enabled entities in some processor, it becomes idle as shown for processors 2-3, 0-3 and 0-1 in Figure 1 respectively. To our knowledge, it is the first use of dynamic deactivation technique to treat multiphase flows in parallel.

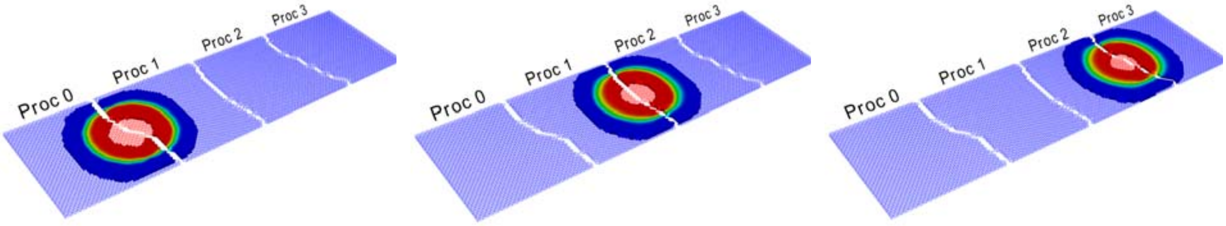


Figure 1 – Parallel dynamic deactivation in the advection of a circle.

5 TEST PROBLEM

This section aims the discussion of the test problem employed to evaluate the performance and robustness of the proposed scheme regarding solution correctness, mass conservation of the discretization method and the parallel dynamic deactivation performance. All computations were carried out on a SGI Altix 350 system, with 6 nodes (SGI C-bricks), each one with 2 Intel Itanium-2 with 1.5 GHz and 4 GB of memory per node; the memory provided by each node is shared through the NUMA link summarizing 24 GB of global system memory. The system runs Red Hat Linux Advanced Server, Intel Fortran compiler 8.1, and SGI Message Passing Toolkit (MPT 1.10.1). No optimizations besides those provided by standard compiler flags (-O3) were used.

5.1 Dam Break Problem

The collapse of a water column is a well known problem, widely employed to validate free-surface codes based on interface capturing methods, since it has experimental results (see [1, 2] for details) and various numerical results from different methods available (see for instance [3]). Furthermore, this problem presents regions with breaking waves and fluid fragmentation. This problem consists of a water column initially sustained by a dam which is suddenly removed. The water falls under the influence of gravity (g = 9.81 m/s^2), acting vertically, and flows downward until hitting the opposite wall producing a sloshing effect. The model, as shown in Figure 2, is simply a box with dimension 4a × a × 2.4a, where a is a parameter, assumed to be equal to 0.146 m, following reference [1]. The water column has dimensions a × a × 2a. The unstructured mesh was built with 46,766 nodes, 251,807 tetrahedra and 306,597 edges. The density of water is ρ_w=1000 kg/m^3 and the dynamic viscosity μ_w = 0.01 kg/(m s). The density of the air was assumed to be ρ_a=1 kg/m^3 and the dynamic viscosity μ_a = 0.0001 kg/(m s).

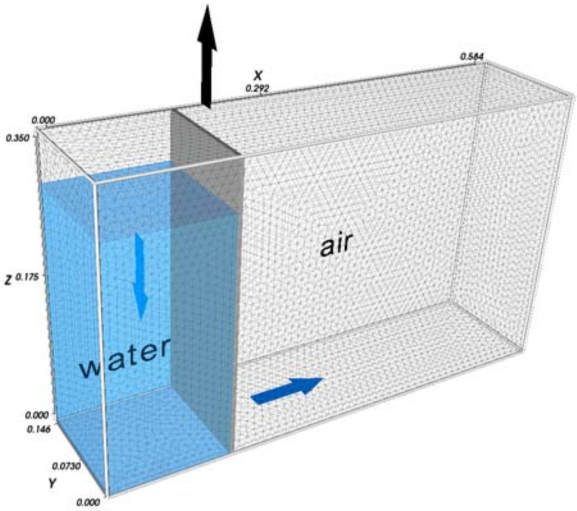


Figure 2 – Model for the collapse of a water column problem.

The validation results accessed from the position of the water column leading edge plotted against the dimensionless time were compared with experimental results as shown in Figure 3.

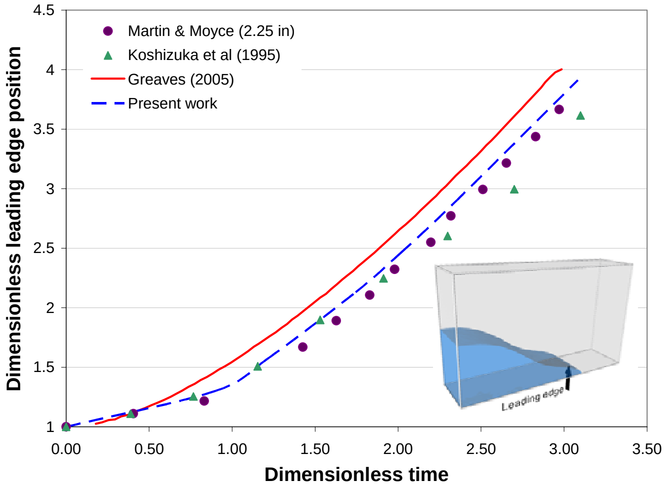


Figure 3 – Validation of the leading edge position for the dam break problem.

Figure 3 shows that the proposed scheme reached good agreement in predicting the position of the water column leading edge, compared to experimental and numerical reference results

[1, 2, 3]. Moreover, from the snapshots shown in Figure 4 (a) and (b) for the simulation instants 0.2 and 0.4 sec, we can qualitatively compare the results obtained with our code with those presented by Koshizuka *et al* in [1] for their experiment.

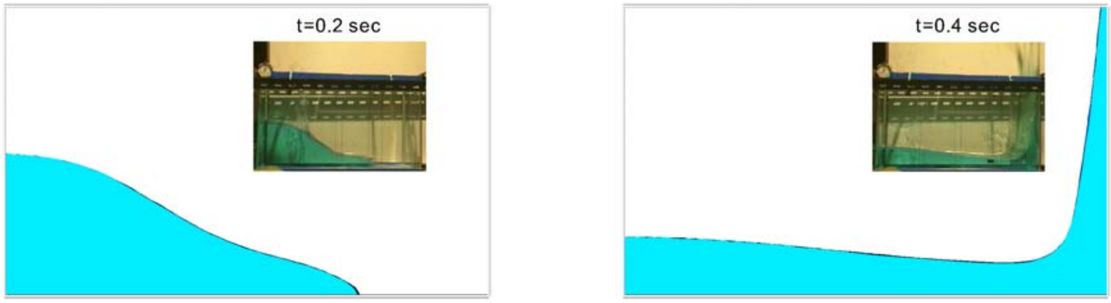
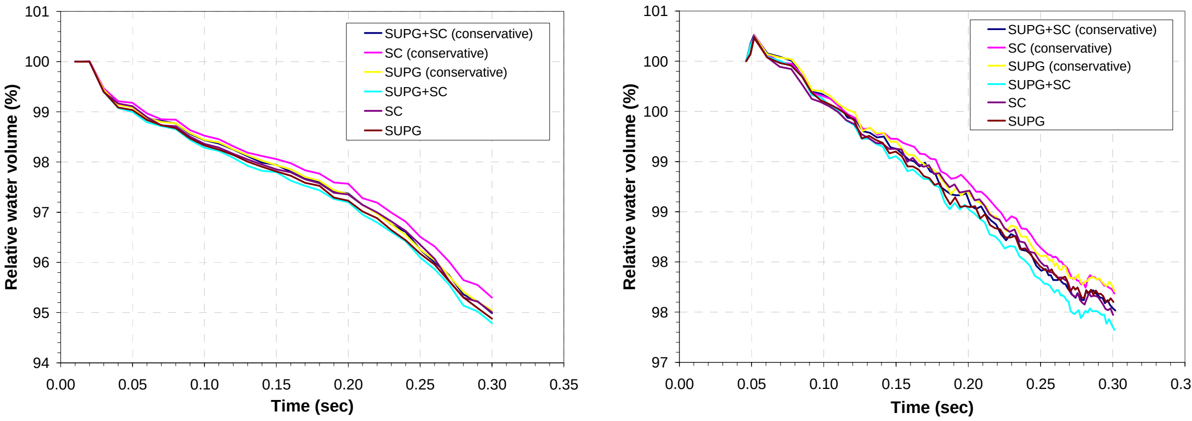


Figure 4 – Snapshots for the simulation instant 0.2 and 0.4 seconds compared with the experimental results presented by Koshizuka [1].

In order to estimate how the stabilized formulation for the marking function influences the volume conservation, we present in Figure 5 (a) and (b) the percentage of volume preserved for the solution scheme employing fixed and adapted time steps with different stabilizations. Moreover, we have evaluated the influence of the Eq. (12) discretized in its conservative and non-conservative forms. Figure 5 (a) presents the results using a fixed time step of 0.01, which comprises a CFL ranging from 0.3 to 16 for this problem. Figure 5 (b) describes the volume conservation when using a time step adjusted with a proportional-integral-derivative controller (PID for short) [5].



(a) Fixed time step (0.01)

(b) Adaptive time step

Figure 5 – Volume preservation according to the time step choice and stabilized formulation adopted.

We can see from Figure 5 that the most conservative scheme in all tests was the one using only the shock capturing term acting on the transport equation in its conservative form. The worst result was obtained by the SUPG and shock capturing stabilizations used simultaneously on the transport equation in a non-conservative form.

The parallel dynamic deactivation algorithm behavior can be observed in Figure 6. This figure shows the amount of active edges during the simulation for a 2 processors run. We should emphasize that, in the PDD algorithm, the amount of any finite element entity (element, edge or node) enabled or disabled has a direct relationship with the computational effort employed to solve the transport equation. Note that the PDD algorithm works enabling and disabling the finite element entities as the solution front advances, following the water column leading edge. At the beginning of the solution procedure, the whole computational effort is spent only on process 0 while process 1 is kept idle. A good load balance is reached only after 0.27 seconds which is lost afterwards. This result suggests that even without a good parallel load balance, the PDD algorithm can be faster due its ability to reduce, considerably, the number of equations following the solution regions with high gradients.

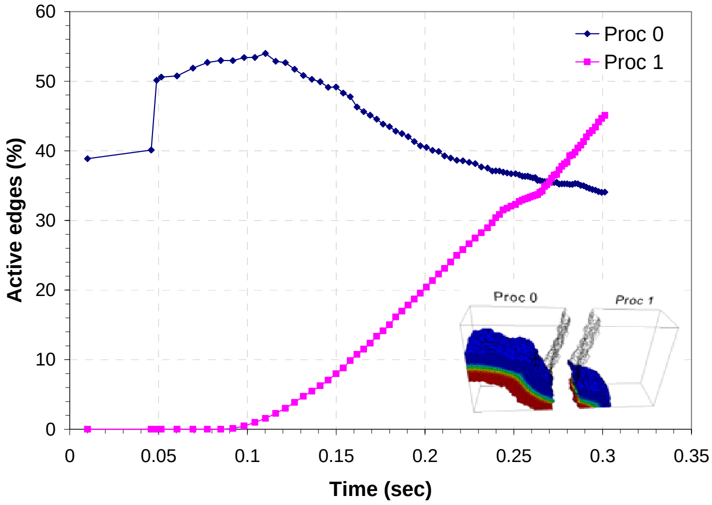


Figure 6 – Dynamic deactivation of the transport solver - percentage of active edges.

6 CONCLUSIONS

This work presented an interface-capturing solution based on the volume-of-fluid method in a finite element context. The three dimensional and unsteady Navier-Stokes equations were discretized with the SUPG/PSPG stabilized finite element formulation while the transport problem was treated with the SUPG and the CAU shock capturing stabilization. The turbulence effects were considered in a large eddy simulation context by a simple classical Smagorinsky model and the time step controlled by a feedback PID strategy. All implementation supported hybrid parallelism as well as vectorization and pipelining common in HPC systems. The main computational kernels were optimized with an edge-based data structure and the computational efforts in the interface-capturing solution were further optimized by the use of the parallel dynamic deactivation technique. The results of the dam break simulation showed us that the proposed scheme is fast and accurate, with mass losses of less than 3% in the dam break problem. Further computational gains can be reached by the use of the dynamic deactivation scheme on the Navier-Stokes solution.

Acknowledgements

The authors would like to thank the financial support of the Petroleum National Agency (ANP, Brazil) and MCT/CNPq, The Brazilian Council for Scientific Research. The Center for Parallel Computations (NACAD) and Laboratory of Computational Methods in Engineering (LAMCE) at the Federal University of Rio de Janeiro provided the computational resources for this research. Dr Marcos Martins is supported by CNPq through CT-PETRO/PROSET 500.196/02-8 grant.

REFERENCES

1. Koshizuka, S, Tamako H and Oka, Y, 1995. A Particle Method for Incompressible Viscous Flow with Fluid Fragmentation, *Computational Fluid Mechanics Journal*; 113:134-147.

2. Martin JC, Moyce WJ, 1952. An Experimental Study of the Collapse of Liquid Columns on a Rigid Horizontal Plane, *Philosophical Transactions of the Royal Society of London,* Series A, 244:312-324.

3. Greaves DM, 2005. Simulation of viscous water column collapse using adapting hierarchical grids, *International Journal for Numerical Methods in Fluids,* 50(6): 693-711.

4. Hirt CW and Nichols BD, 1981. Volume of Fluid (VOF) Methods for the Dynamics of Free Boundaries, *Journal of Computational Physics*, (39)201-225.

5. Valli, A.M.P., G. F. Carey and A.L.G.A. Coutinho, 2005. Control strategies for timestep selection in FE simulation of incompressible flows and coupled reaction-convection-diffusion processes, *Int. J. Num. Meth. in Fluids*, 47, 201-231.

6. Lohner R and Camelli F., 2004. Dynamic Deactivation for Advection-Dominated Contaminant Transport, *Communications in Numerical Methods in Engineering*, 20:639-646.

7. Souza, DAF, Martins MAD and Coutinho ALGA, 2005. Edge-Based Adaptive Implicit/Explicit Finite Element Procedures for Three-Dimensional transport problems, *Communications in Numerical Methods in Engineering*; 21:545-552.

8. Galeão AC, do Carmo EGD, 1988. A Consistent Approximate Upwind Petrov-Galerkin Method for Convection-Dominated Problems, *Computer Methods in Applied Mechanics and Engineering* 68 (1): 83-95.

9. Brooks, A. N. and Hughes, T. J. R., 1982. Streamline Upwind/Petrov-Galerkin Formulations for Convection Dominated Flows with Particular Emphasis on the Incompressible Navier-Stokes Equation, *Comput. Methods Appl. Mech. Engrg.* 32: 199-259.

10. Tezduyar, T. E., 1999. Finite Elements in Fluids: Lecture Notes of the Short Course on Finite Elements in Fluids, *Computational Mechanics Division* – Vol. 99-77, Japan Society of Mechanical Engineers, Tokyo, Japan.

11. Tezduyar, T. E., 1991. Stabilized Finite Element Formulations for Incompressible Flow Computations, *Advances in Applied Mechanics,* 28: 1-44.

12. Tezduyar, T. E., Mittal S., Ray S. E. and Shin R., 1992. Incompressible Flow Computations with Stabilized Bilinear and Linear Equal-Order-interpolation Velocity-pressure Elements, *Comput. Methods Appl. Mech. Engrg.* 95: 221-242.

13. Coutinho ALGA, Martins MAD, Sydenstricker RM and Elias RN, 2006. Performance Comparison of Data-reordering Algorithms for Sparse Matrix-vector Multiplication in Edge-based Unstructured Grid Computations, *International Journal for Numerical Methods in Engineering*, 66:431-460.

14. Franca LP and Frey SL, 1992. Stabilized Finite Element Methods: II. The Incompressible Navier-Stokes Equations, *Computer Methods in Applied Mechanics and Engineering*, 99:209-233.

15. Elias RN, Coutinho ALGA and Martins MAD, 2006. Inexact Newton-type Methods for the Solution of Steady Incompressible Viscoplastic Flows with the SUPG/PSPG Finite Element Formulation, *Computer Methods in Applied Mechanics in Engineering*; 195:3145-3167.

16. Elias RN, Martins MAD and Coutinho ALGA, 2005. Parallel Edge-Based Inexact Newton Solution of Steady Incompressible 3D Navier-Stokes Equations, *Proceeding of the 11th International Euro-Par Conference on Lecture Notes in Computer Science 3648*, pp.1237-1245, Lisbon, Portugal;

17. Kleefsman, K. M. T., 2005. Water Impact Loading on Offshore Structures: A Numerical Study. *PhD Thesis*, University of Groningen.

18. Tezduyar T.E., 2003. Computation of moving boundaries and interfaces and stabilization parameters, *Int. J. Numer. Meth. Fluids*, 43:555–575.

19. Lohner R., Yang C., Onate E. , 2006. On the simulation of flows with violent free surface motion. *Computer Methods in Applied Mechanics and Engineering*, in press.

20. Martins MAD, Elias RN and Coutinho ALGA, 2006. EdgePack: A parallel vertex and node reordering package for optimizing edge-based computations in unstructured grids, *submitted to the Seventh International Meeting on High Performance Computing for Computational Science – VECPAR*, Rio de Janeiro, Brazil