PARALLEL BLOCK-STRUCTURED ADAPTIVE MESH REFINEMENT FOR INCOMPRESSIBLE NAVIER-STOKES EQUATIONS

Rafael Sene de Lima, rafaelsene@gmail.com
Aristeu da Silveira Neto, aristeus@mecanica.ufu.br
Millena Martins Villar, millena.villar@gmail.com
Márcio Ricardo Pivello, pivello@gmail.com
Universidade Federal de Uberlândia, Dept. of Mechanical Engineering, Uberlândia - Brazil

Alexandre Megiorin Roma, roma@ime.usp.br
Universidade de São Paulo. Dept. of Applied Mathematics. São Paulo - Brazil

Ricardo Serfaty, rserfaty@petrobras.com.br
Petrobras - Cenpes. Equipment management. Rio de Janeiro - Brazil

Abstract. *The numerical simulation of fluid flow involving complex geometries is greatly limited by the spatial grid resolution required. These flows often contain small regions with complex motions, while the remaining flow is relatively smooth. Adaptive mesh refinement (AMR) enables the spatial grid to be refined in local regions that require finer grids to resolve the flow. This work describes an approach to parallelization of a structured adaptive mesh refinement (SAMR) algorithm. This type of methodology is based on locally refined grids superimposed on coarser grids to achieve the desired resolution in numerical simulations. The main elements to achieve parallelization of SAMR algorithms are a dynamic load-balancing method to distribute work to processors and a dynamic distribution technique to manage communications. The methodology is based on a message passing model using the recursive coordinate bisection (RCB) for domain partition. For this work, a semi-implicit projection method has been implemented to solve the incompressible Navier-Stokes equations.*

Keywords: *Adaptive Mesh Refinement, Dynamic Load Balancing, Incompressible Navier Stokes Equations.*

1. INTRODUCTION

Over the past years, the improvement of computational methods for solving the incompressible Navier-Stokes equations for fluid flow problems, the increase in computing capacity and the advent of parallel computer, have proven to be very useful for better understanding the physics of various fluid flows of increased complexity. However, the required mesh resolution to accurately resolve small scale fluid motions at specific regions containing large gradients, such as shocks and interfaces, combustion and near-wall regions of complex boundaries, remains as a limitation for these methods.

One approach to relieve the issue of mesh resolution is called Adaptive Mesh Refinement (AMR). Adaptively refining and coarsening local regions of the computational domain results in varying levels of spatial grid resolution, which are locally determined by accuracy requirements. Methods based on SAMR start with a coarse base grid with minimum acceptable resolution that covers the entire computational domain. As the solution progresses, regions in the domain with unacceptable solution errors, requiring higher resolution, are identified and refined. Refinement proceeds recursively so that the refined regions requiring higher resolution are similarly tagged and even higher resolution grids are overlaid on these regions (Berger and Colella, 1989). The result is a dynamic adaptive grid hierarchy, as shows Fig. 1. Parallel implementations of SAMR methods offer the potential for accurate simulations of high complexity fluid flows. However, they present interesting challenges in dynamic resource allocation, data-distribution and load-balancing. The overall

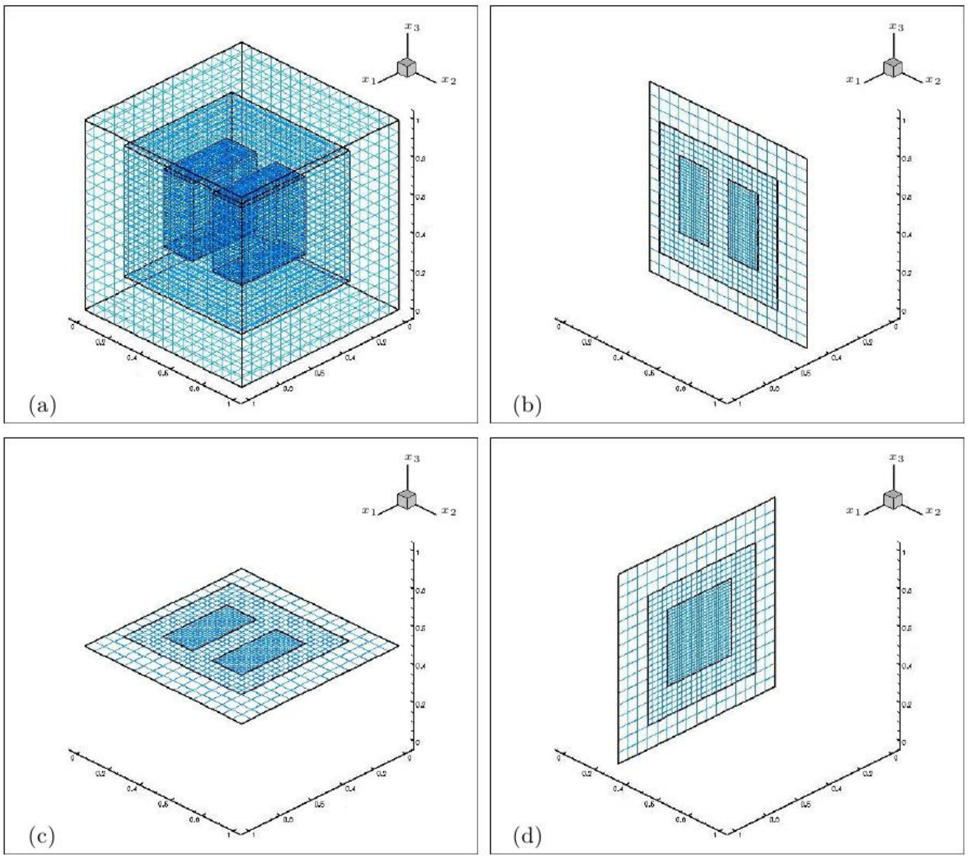


Figure 1. Block-structured mesh composed by three refinement levels in a computational domain $[0,1] \times [0,1] \times [0,1]$. (a) Three-dimensional mesh; (b) slice in $x = 0.42$; (c) slice in $z = 0.45$; (d) slice in $y = 0.3$ Nós (2007).

efficiency of parallel SAMR applications is limited by the ability to partition the underlying grid hierarchies at run-time to expose all inherent parallelism, minimize communication and synchronization overheads, and balance load.

This work presents an approach to parallelization of block-structured adaptive mesh refinement for the incompressible Navier-Stokes equations. The domain partition and load balance are performed using the Recursive Coordinate Bisection method (RCB) implemented on Zoltan load balancing library (Devine *et al.*, 2002).

2. METHODOLOGY

The flow of an incompressible fluid is governed by mass and momentum conservation laws, which take the form of the Navier-Stokes equations

$$\rho\left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u}\right) = -\nabla p + \mu\nabla^2\mathbf{u} + \rho\mathbf{g}, \tag{1}$$

$$\nabla\cdot(\mathbf{u}) = 0, \tag{2}$$

where $\mathbf{u}$ is the velocity vector, p is the pressure, ρ and μ are density and dynamic viscosity, respectively and g represents the gravitational acceleration. The projection method applied to solve the Eq. (1) and Eq. (2) splits the time advancement of the variables into two main steps: advancing an approximate velocity field, and then projecting that field onto the divergence free field. An approximation to the velocity field, $\tilde{\mathbf{u}}$, is obtained using a semi-implicit temporal discretization method, used by (Villar, 2007) and described by (Ascher *et al.*, 1995):

$$\frac{\rho}{\Delta t}\left(\alpha_2\tilde{\mathbf{u}}^{n+1} + \alpha_1\tilde{\mathbf{u}}^n + \alpha_0\tilde{\mathbf{u}}^{n-1}\right) = \\ \lambda\nabla^2\tilde{\mathbf{u}}^{n+1} + \beta_1 h_1(\tilde{\mathbf{u}}^n,\mu) + \beta_0 h_0(\tilde{\mathbf{u}}^{n-1},\mu) - \nabla p^n + \rho\mathbf{g}, \tag{3}$$

where

$$h = -\lambda\nabla^2\mathbf{u} + \nabla\cdot\left[\mu(\nabla\mathbf{u} + \nabla\mathbf{u}^T)\right] - \mathbf{u}\cdot\nabla\mathbf{u}, \tag{4}$$

$$\lambda = C_\lambda \parallel \mu \parallel_\infty . \tag{5}$$

In this paper, we adopt $C_\lambda = 2$. This choice is based in the work of (Ceniceros *et al.*, 2010), which were guided by their own Numerical experiments and by the recent work of (Xu and Tang, 2006).

To resolve the Eq. (1) and Eq. (2), we still need to define the elliptic partial differential equation, or Poisson equation to calculate the pressure correction q,

$$\nabla\cdot\left(\frac{1}{\rho}\nabla q^{n+1}\right) = \frac{\alpha_2}{\Delta t}\nabla\cdot\tilde{\mathbf{u}}^{n+1}, \tag{6}$$

where the Eq. (3) and Eq. (6) should be solved using the Multilevel Multigrid Method.

2.1 Load Balance

The main function of parallel computing is to distribute data among processors. However, when AMR is involved to adaptively increase the resolution of the grid only where it is needed, to reduce the workload, the mesh keeps changing in computation. In this situation a dynamic load balancing algorithm needs to be called every time the mesh is refined. This work uses the Recursive Coordinate Bisection method (RCB) for load balance and domain partition. The RCB method was first proposed as a static load-balancing algorithm by (Berger and Bokhari, 1987), but is also attractive as a dynamic load-balancing algorithm for AMR because it implicitly produces incremental partitions according to the mesh refinement. The algorithm chooses a direction and then split the mesh by making an appropriate perpendicular cut. The position of the cut should be such that an equal number of mesh elements fall on either side of it. The sub-domains are then further divided by recursive application of the same splitting algorithm until the number of partitions reaches the number of processors.

The load balancing algorithm implemented in this work uses the RCB method of Zoltan load balancing library proposed by Devine *et al.* (2002). Zoltan is a toolkit for parallel load balancing and data management in scientific computing developed by Sandia National Laboratories. It contains a collation of load balancing algorithms. Zoltan also supports Graph, Hypergraph partitioners, some of them implemented by Zoltan itself and others are from some famous third party load balancing library. These features helped Zoltan to build an unified interface, so that the users can easily switch from one load balancing algorithm to another algorithm.

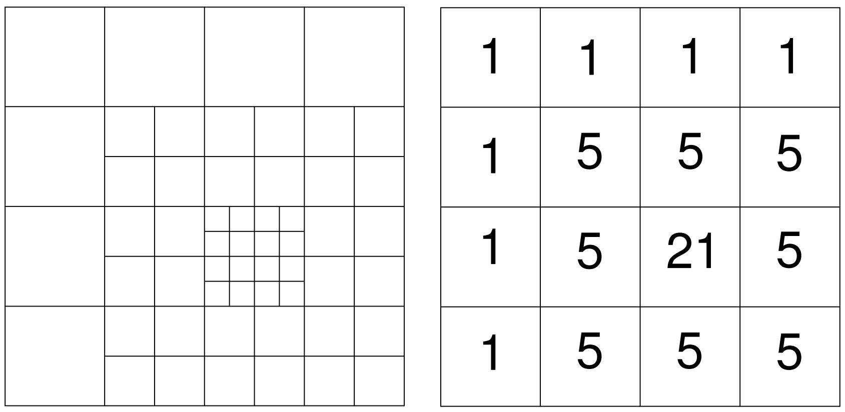


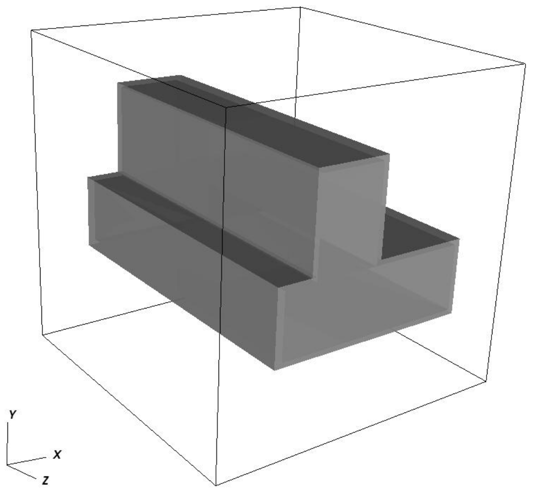
Figure 2. Weights distribution over an adaptive mesh hierarchy.

3. RESULTS

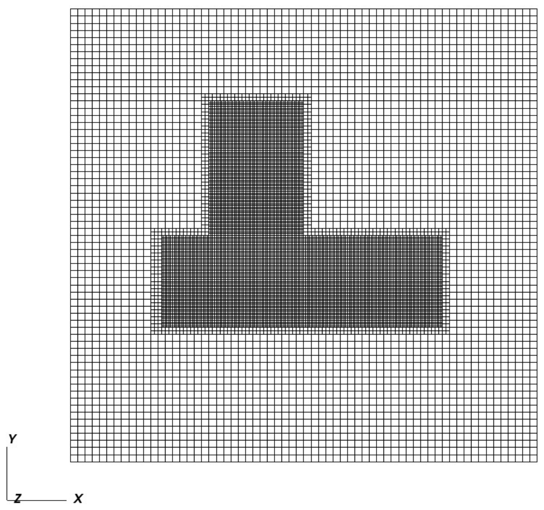
To evaluate both the parallel version of the implementation of the Multilevel Multigrid Method and the parallel version of the projection method, we perform a convergence test for the Navier-Stokes equations based on the Method of Manufactured Solutions. For the validation of the proposed methodology, we simulated the case of a sliding lid cavity for Reynolds numbers 100, 400 and 1.000.

3.1 Verification Methodology

The convergence test is performed with load balance, 8 processors and a 32^3 base level grid with two refinement levels, using Dirichlet boundary conditions for pressure p and Neumman boundary conditions for the velocity field $\mathbf{u}$. Fig. 3 shows the composite mesh used in the numerical convergence test. For this test, the Navier-Stokes equations for an



(a)



(b)

Figure 3. Composite mesh $(32L3)$, used in the numerical convergence test. (a) Perspective view, (b) cross section view at $z = 0.5$.

incompressible fluid flow are

$$\rho\Big(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u}\Big) = -\nabla p + \nabla\cdot\big[\mu\big(\nabla\mathbf{u} + \nabla\mathbf{u}^T\big)\big] + \mathbf{f}, \tag{7}$$

where $\mathbf{f}$ is the forcing therm,

$$\mathbf{f} = \rho_e\frac{\partial \mathbf{u_e}}{\partial t} + \rho_e\cdot\mathbf{u_e}(\nabla\mathbf{u_e}) + \nabla p_e - \nabla\cdot(\mu_e\nabla\mathbf{u_e}). \tag{8}$$

The exact solutions for the velocity field $\mathbf{u}_e$ for the pressure field p_e, density ρ_e and viscosity μ_e are given by

$$u_e = sen^2(2\pi x + 2\pi y + 2\pi z + t), \tag{9}$$
$$v_e = -cos^2(2\pi x + 2\pi y + 2\pi z + t), \tag{10}$$
$$w_e = 2cos^2(2\pi x + 2\pi y + 2\pi z + t), \tag{11}$$
$$p_e = cos(2\pi x + 2\pi y + 2\pi z + t), \tag{12}$$
$$\rho_e = 1 + sen^2(2\pi x), \tag{13}$$
$$\mu_e = 1 + 0,2cos^2(2\pi x + 2\pi y + 2\pi z + t). \tag{14}$$

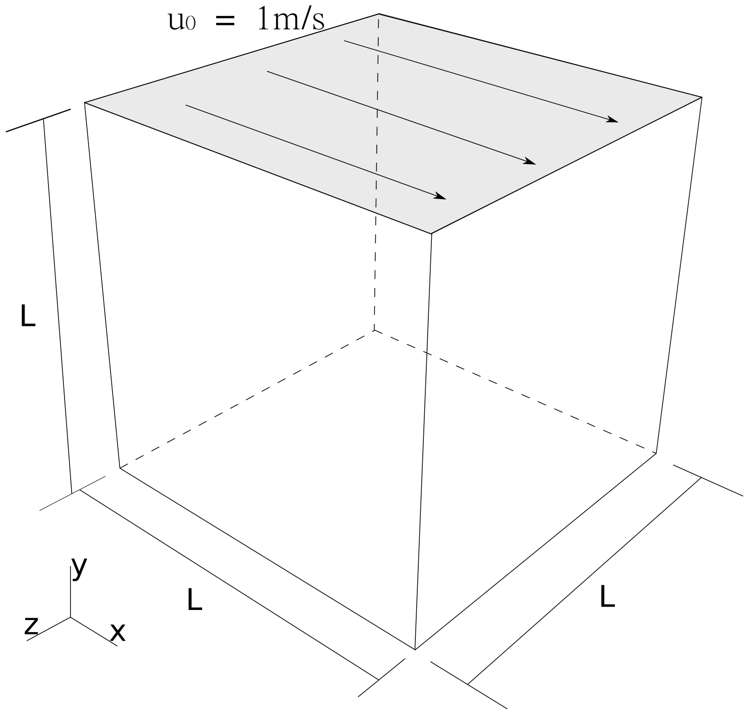
where the exact solutions for u_e, v_e, w_e should be chosen to ensure that $\nabla\mathbf{u} = 0$. The Tab. 3.1shows that all velocities u, v, w reach second order for spatial convergence order, and the pressure p reaches first order for spatial convergence order.

Table 1. Parallel convergence test for Navier Stokes equations.

Mesh	p		w		v		w	
	$\|\phi - \phi_e\|_2$	r_e	$\|\phi - \phi_e\|_2$	r_e	$\|\phi - \phi_e\|_2$	r_e	$\|\phi - \phi_e\|_2$	r_e
16^3L2	0,581	-	1,57E-2	-	1,50E-2	-	2,50E-2	-
32^3L2	0,195	2,98	4,05E-3	3,88	4,01E-3	3,74	6,92E-3	3,61
64^3L2	0,064	3,04	1,05E-3	3,85	1,03E-3	3,89	1,84E-3	3,75

3.2 Lid Driven Cavity Flow

For the validation of the proposed methodology, we simulated the case of a sliding lid cavity for Reynolds numbers 100, 400 and 1.000. The dimensions of the computational domain are $1m \times 1m \times 1m$, the sliding velocity of the lid is $u_0 = 1m/s$ and the density $\rho = 1kg/m^3$. The viscosity is calculated using the Reynolds number, ie $\mu = u_0\rho L/Re$, where L is the cavity length. The cases for $Re = 100$ and $Re = 400$ were performed using 16 processors and a 16^3 base level grid with 3 refinement levels. For $Re = 1000$, the simulation was performed using 32 processors, 32^3 base level grid with two refinement levels. The obtained results are compared with those from (Deshpande and Milton, 1998) for $Re = 1000$ and with those of (Ku *et al.*, 1987) for $Re = 100$ and $Re = 400$. The comparisons are carried out by means of the velocity u along the vertical axis (y) of the cavity at $x/L = 0.5$ and $z/L = 0.5$ and velocity v along the horizontal axis (x) of the driven cavity at $y/L = 0.5$ and $z/L = 0.5$. Fig. 4 shows a representative sketch of a three-dimensional sliding lid cavity.



(a)

Figure 4. Representative sketch of a three-dimensional sliding lid cavity

Figures 5 and 6 show u and v velocity profiles for $Re = 100$ and $Re = 400$, which present good agreement with

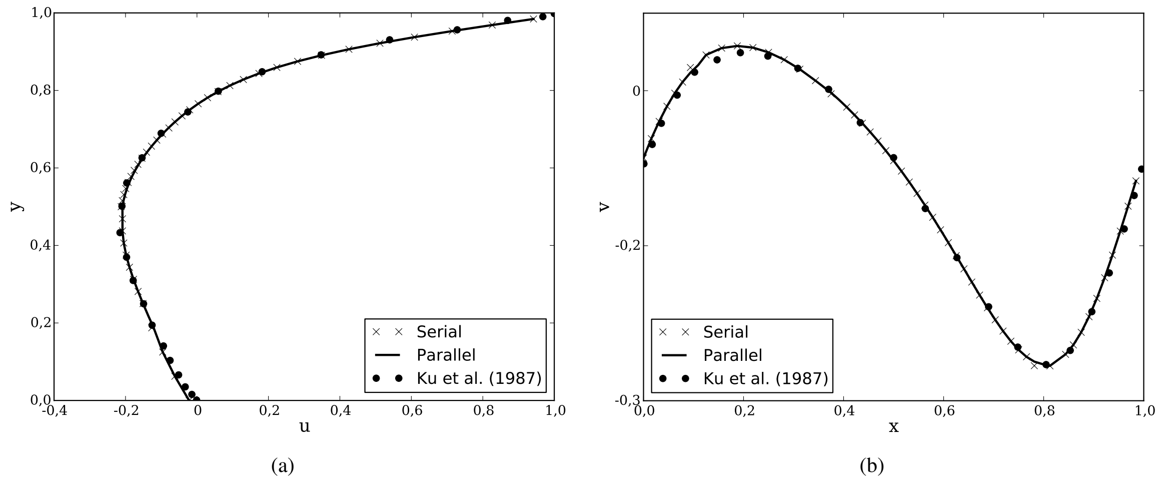


Figure 5. Comparison of velocity profiles in a composite mesh $16L3$ using 16 processors for $Re = 100$. (a) $u(y)$ at $x/L = 0.5$ and $z/L = 0.5$; (b) $v(x)$ at $y/L = 0.5$ e $z/L = 0.5$.

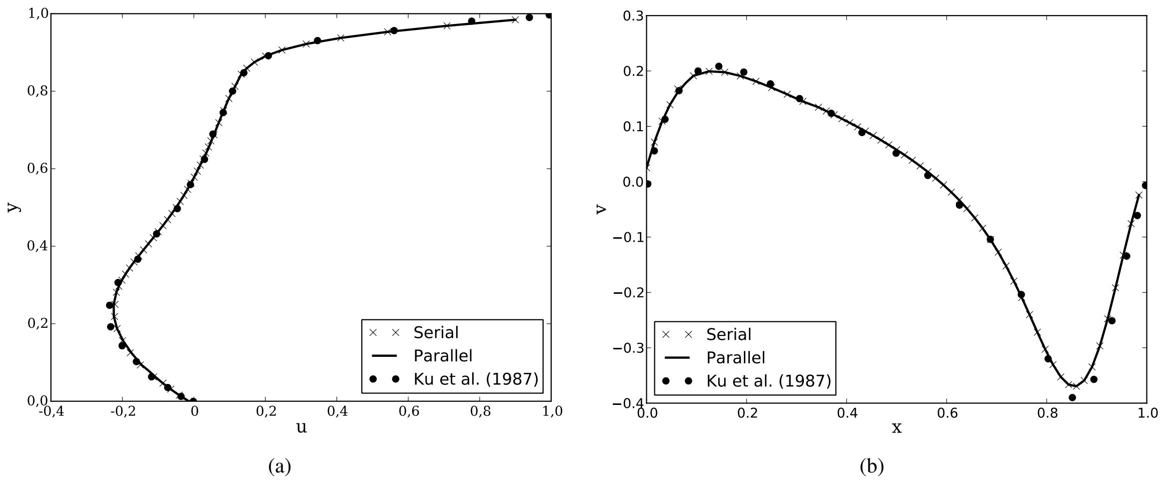
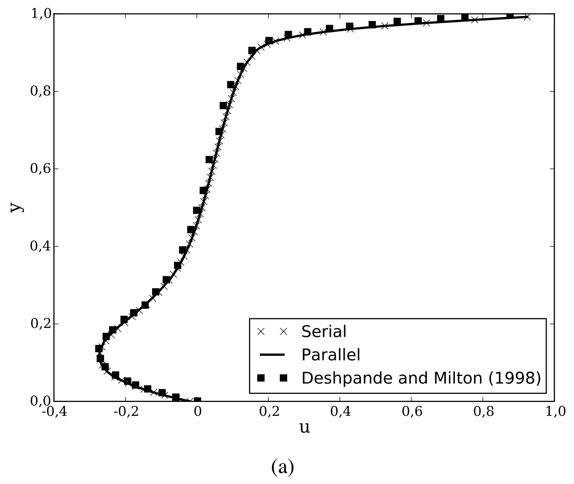


Figure 6. Comparison of velocity profiles in a composite mesh $16L3$ using 16 processors for $Re = 400$. (a) $u(y)$ at $x/L = 0.5$ and $z/L = 0.5$; (b) $v(x)$ at $y/L = 0.5$ e $z/L = 0.5$.

the numerical results of (Ku *et al.*, 1987). The simulation time for $Re = 100$ was of 33 minutes in parallel and 300 minutes in serial, leading to a gain of approximately 9 times in the total time of simulation. The simulation for $Re = 400$ spent 40 minutes in parallel and 420 minutes in serial, which represents a gain of approximately 10 times in the total time of simulation. The Fig. 7 shows the u and v velocity profiles for $Re = 1000$, which present good agreement with the numerical results of (Deshpande and Milton, 1998).

Figure 8 shows the number of computational cells at each processor for 4 time steps. There was an increase in the number of cells of all time steps, because in the early stages of the simulation, regions of high vorticity are concentrated in the upper region of the cavity, forcing the adaptive mesh to concentrates at this location. Generally, there is a reasonable distribution of the computational load throughout the simulation, however, in some cases the difference between the number of cells is up to 50%, which degenerates parallel performance.



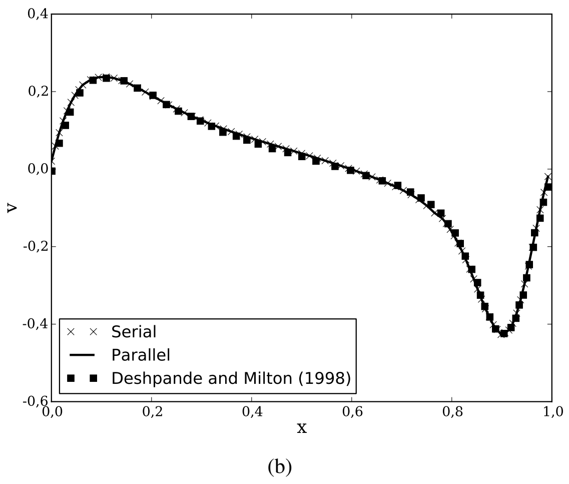


Figure 7. Comparison of velocity profiles in a composite mesh $32L3$ using 32 processors for $Re = 1000$. (a) $u(y)$ at $x/L = 0.5$ and $z/L = 0.5$; (b) $v(x)$ at $y/L = 0.5$ e $z/L = 0.5$.

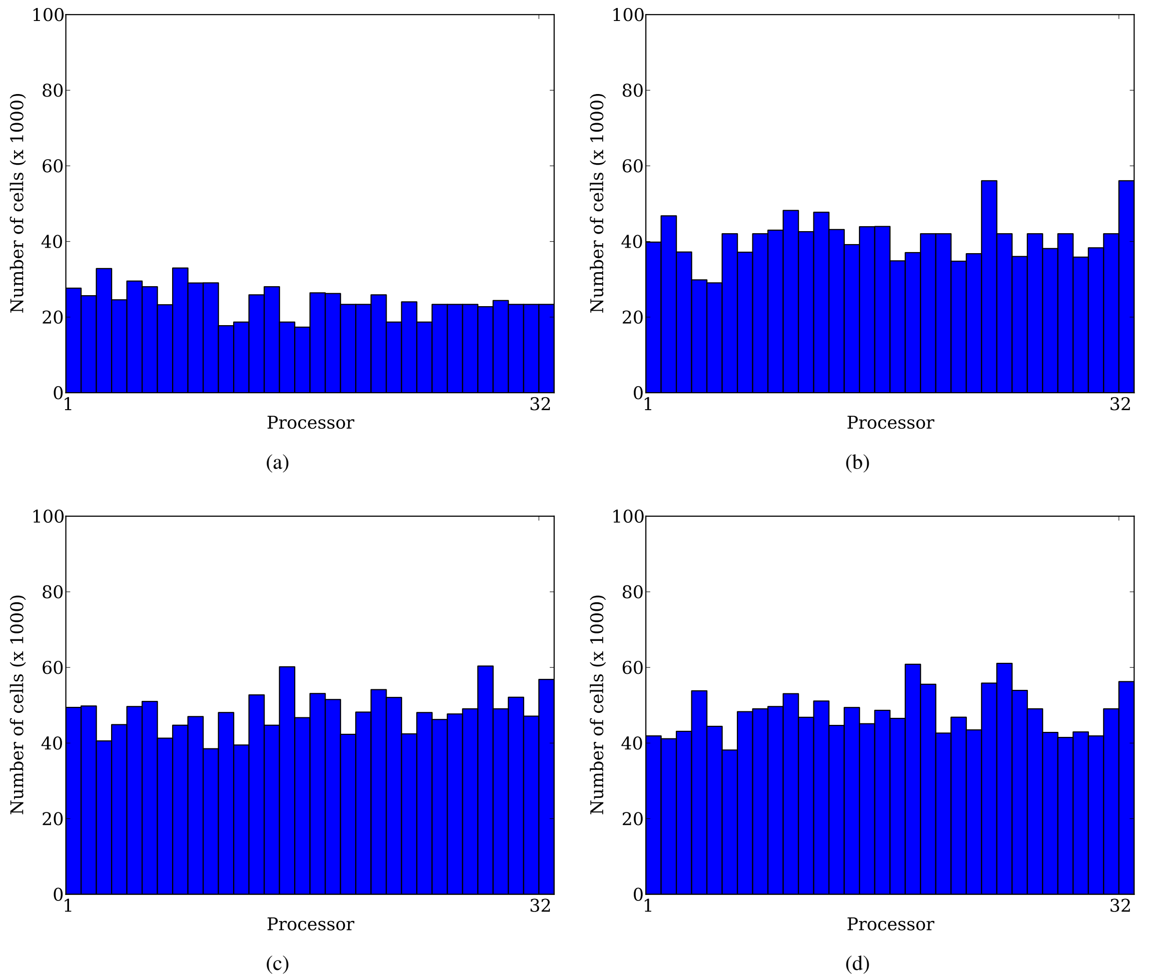


Figure 8. Distribution of computational cells in 32 processors for a composite mesh $32L3$ for $Re = 1000$. (a) 12s; (b) 23s; (c) 35s; (d) 47s.

4. CONCLUSION

A parallelization approach, with dynamic load balancing, using the Zoltan library was implemented for a 3D numerical code based on the SAMR methodology. Preliminary results shows a stable behavior and the maintenance of the characteristic of the original code: second order of convergence for the velocities and the same accuracy.

5. ACKNOWLEDGEMENTS

The authors would like to thanks to the Universidade Federal de Uberlândia, CNPq $309433/2011-8$, CNPq $\#140874/2008$, FAPEMIG and PETROBRAS for supporting the present work.

6. REFERENCES

Ascher, U.M., Ruuth, S.J. and Wetton, B.T.R., 1995. “Implicit-explicit methods for time-dependent partial differential equations”. *SIAM Journal on Numerical Analysis*, Vol. 32, pp. 797–823.

Berger, M.J. and Colella, P., 1989. “Local adaptive mesh refinement for shock hydrodynamics”. *Journal of Computational Physics*, Vol. 82, pp. 64–84.

Berger, M. and Bokhari, S., 1987. “A partitioning strategy for nonuniform problems on multiprocessors”. *IEEE Transactions on Computers*, Vol. 36, pp. 570–580. ISSN 0018-9340.

Ceniceros, H.D., Roma, A.M., Silveira-Neto, A. and Villar, M.M., 2010. “A robust, fully adaptive hybrid level-set/front-tracking method for two-phase flows with an accurate surface tension computation”. *Communications in Computational Physics*, Vol. 8, pp. 51–94.

Deshpande, M.D. and Milton, S.G., 1998. “Kolmogorov scales in a driven cavity flow”. *Fluid Dynamics Research*, Vol. 22, No. 6, p. 359. URL http://stacks.iop.org/1873-7005/22/i=6/a=A04.

Devine, K., Boman, E., Heapby, R., Hendrickson, B. and Vaughan, C., 2002. “Zoltan data management service for parallel dynamic applications”. *Computing in Science and Engineering*, Vol. 4, pp. 90–97.

Ku, H.C., Hirsh, R.S. and Taylor, T.D., 1987. “A pseudospectral method for solution of the three-dimensional incompressible navier-stokes equations”. *Journal of Computational Physics*, Vol. 70, No. 2, pp. 439 – 462. ISSN 0021-9991. doi:10.1016/0021-9991(87)90190-2. URL http://www.sciencedirect.com/science/article/pii/0021999187901902.

Nós, R.L., 2007. *SimulaÃ§ões de Escoamentos tridimensionais Bifásicos Empregando Métodos Adaptativos e Modelos de Campo de Fase*. Ph.D. thesis, Universidade de São Paulo.

Villar, M.M., 2007. *Análise Numérica Detalhada de Escoamentos Multifásicos Bidimensionais*. Ph.D. thesis, Universidade Federal de Uberlândia.

Xu, C. and Tang, T., 2006. “Stability analysis of large time - stepping methods for epitaxial growth models”. *SIAM Journal on Numerical Analysis*, Vol. 44, No. 4, pp. 1759–1779. URL http://epubs.siam.org/doi/abs/10.1137/050628143.