On the Efficiency of the Parallel-in-Time Finite Volume Calculation of the Unsteady Navier-Stokes Equations

J. M. F. Trindade[1] **and J. C. F. Pereira**[2]

Abstract: In this paper, we discuss the efficiency and speed-up of parallel-in-time calculations of the unsteady incompressible Navier-Stokes equations in a PC-cluster. The parallel-in-time method is based on the alternate use of coarse global sequential solvers with fine local parallel ones in an iterative predictor-corrector fashion. Therefore, the efficiency of parallel calculations is strongly dependent on the number of iterations required for convergence. The one-dimensional scalar transport equation and the two-dimensional incompressible unsteady form of the Navier-Stokes equations were used to conduct numerical experiments to derive some conclusions concerning the accuracy and convergence of the iterative method. A simple performance model is proposed to estimate the efficiency of the parallel calculations as a function of the most relevant parameters that contribute to the computing time required to perform a parallel-in-time calculation. Among them, we have analyzed the influence of the number of processors, the number of iterations in the parallel-in-time algorithm and the influence of the coarse to fine time-grid step size ratio. The good agreement between the obtained parallel efficiency and the values estimated by the proposed performance model allows to conclude that parallel-in-time efficiency is quantitatively different from the parallel efficiency of the space domain decomposition, regarding the number of processors available for a fixed problem dimension. A significant speed-up is possible when the temporal scale of the problem is large and enough processors are available.

Keyword: parallel-in-time, parallel efficiency, Navier-Stokes, finite volume.

1 Introduction

The parallelism in the time direction is not common in Computational Fluid Dynamics. Parabolic and hyperbolic differential equations are usually solved numerically by algorithms that are sequential in time, see e.g. the simulations reported by Nicolás and Bermúdez (2004) and Shu, Ding, and Yeo (2005). Space domain splitting and the allocation of each sub-domain to a processor is the methodology usually used to perform the parallel computation of the governing fluid flow equations, see e.g. Grimaldi, Pascazio, and Napolitano (2006). At each time step, the processors need to exchange boundary variable values with processors holding adjacent sub-domains. For a fixed space sized problem, in a distributed memory parallel computer the communication/computation ratio increases with the number of processors yielding a decrease on the parallel efficiency. It is believed that in the near future massive parallel computer systems will increase the number of processors available allowing new boundaries to the problems dimension. Consequently, time and hybrid (space and time) domain decomposition methods will have a high potential application to reduce the computing time that nowadays is only achieved with the standard spatial domain decomposition technique. This will have a positive impact on the solution of partial differential equations that CFD deals with but also in other areas of computer modeling of engineering and science.

[1] Escola Náutica Infante D. Henrique, Av. Bonneville Franco, 2780-572 Paco d'Arcos, Portugal. (jorgetrindade@enautica.pt)

[2] Instituto Superior Técnico, Universidade Técnica de Lisboa, Av. Rovisco Pais, 1049-001 Lisboa, Portugal. (jcfpereira@mail.ist.utl.pt)

Lions, Maday, and Turinici (2001) presented a new approach to parallelize across the time domain of the problem under consideration. Bal and Maday (2002) introduced some changes on the original algorithm in order to obtain better stability and efficiency. The application of the algorithm has been performed for simulation of several kinds of science and engineering problems [Baffico, Bernard, Maday, Turinici, and Zerah (2002); Maday and Turinici (2003); Farhat and Chandesris (2003)]. Previous application of the parallel-in-time method for the solution of the unsteady incompressible Navier-Stokes equations indicates that for long time simulations this method can be a promising technique [Trindade and Pereira (2004)]. However, parallel-in-time solution of non-linear fluid flow equations is still in the beginning and many theoretical, numerical and practical topics need to be investigated. Among others, the robustness, the efficiency and the solution accuracy need to be considered for non-linear complex unsteady flows. Parallel-in-time Navier-Stokes solutions were addressed by Trindade and Pereira (2006) providing some guidance to the appropriate choice of the numerical schemes.

The prediction of the theoretical parallel efficiency or speed-up is relevant to decide when to use parallel-in-time for a specific problem solution. The purpose of the present work is to present a simplified performance model for the parallel-in-time method. This model takes into account several parameters that contribute to the computing time required to perform a parallel-in-time calculation. Another important issue related with the performance of the parallel-in-time method is the time spent by the communication tasks required by the algorithm. The performance model is validated with solutions of the parabolic unsteady Navier-Stokes equations. A finite volume approach is used for discretization of the Navier-Stokes equations. The numerical simulation of flow past a two-dimensional square cylinder between parallel walls at a Reynolds number of 500 was selected to illustrate and analyze the properties of the parallel-in-time method through numerical experiments. The comparison of the observed parallel-in-time performance for the solution of a complex unsteady incompressible flow problem on a PC-cluster with the performance model prediction will allow to verify the effect of the communication time on the parallel-in-time simulation efficiency.

In Section 2, we briefly describe the numerical parallel-in-time method. Numerical experiments allow to analyze the convergence of the parallel-in-time algorithm in Section 3. The presentation of a simplified performance model and validation through numerical experiments is included in Section 4 and in the last Section summarizing conclusions are provided.

2 Numerical method

For an incompressible Newtonian fluid and unsteady flow, the governing continuity and Navier-Stokes equations are integrated in each finite control volume and after application of Gauss theorem read as:

$$\int_S \mathbf{v.n}\, dS = 0, \tag{1}$$

$$\frac{\partial}{\partial t}\int_\Omega u_i\, d\Omega + \int_S u_i \mathbf{v}\,.\,\mathbf{n}\, dS = \int_S \gamma\, grad\, u_i\,.\,\mathbf{n}\, dS - \frac{1}{\rho}\int_S p\,\mathbf{i}\,.\,\mathbf{n}\, dS \tag{2}$$

where Ω is the volume and S is the surface of an arbitrary control volume, $\mathbf{n}$ is the unit vector normal to S and directed outwards, $\mathbf{v}$ is the velocity vector, u_i are the Cartesian velocity components, ρ is the density, γ is the viscosity and p is the pressure. Let us assume that an accurate enough spatial discretization is selected. Further details of the spatial discretization will be provided in Section 3. The parallel-in-time algorithm [Bal and Maday (2002)] is based on the iterative use of coarse global sequential solvers (or integrators) with fine local parallel ones, allowing the time domain decomposition and the propagation of solution jumps on a coarse time-grid. The scheme combines a very precise simulation run on parallel over a set of non-overlapping time intervals with a coarse simulation over the entire

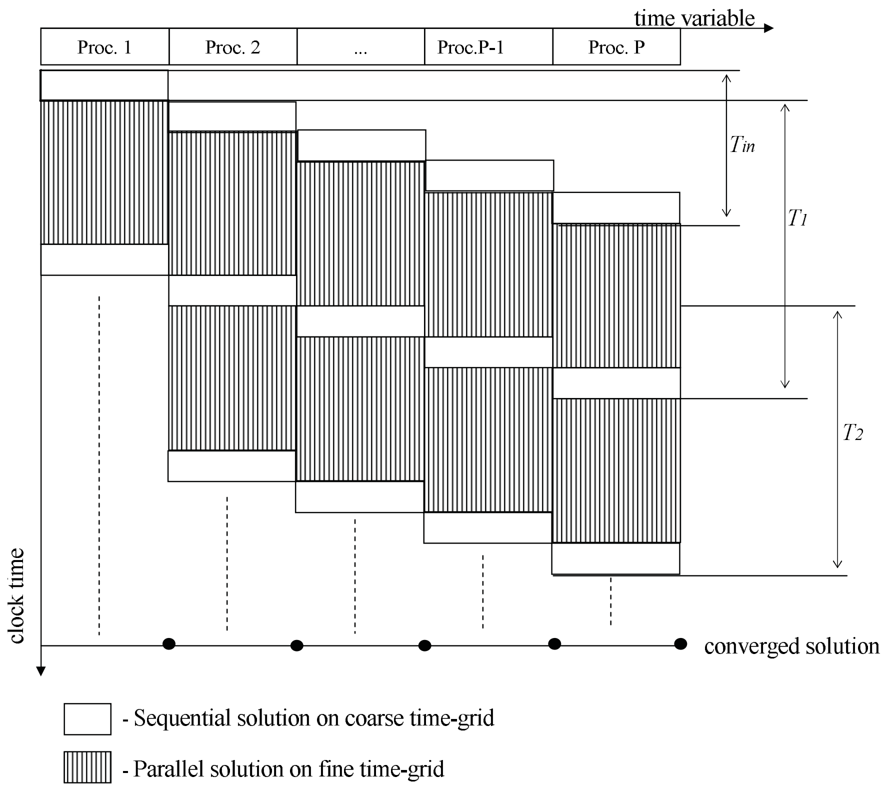

Figure 1: Parallel-in-time solver schematic diagram.

time-span. The diagram in Fig. 1 describes the parallel-in-time methodology. The time variable is represented in abscissas and in ordinates the computing time. Firstly, the time domain $[0,T]$ of the problem under consideration is decomposed into a sequence of P (number of processors) subdomains of size Δt dictating the coarse time-grid resolution. The integrator in the coarse time-grid employs the time-step Δt and the integrator on the finer time-grid uses a smaller interval given by $\delta t = T/M$, for some integer M. The initial and successive variable fields at $t_i = i \times \Delta t$, where i is the processor number, are denoted by u_0 and u_1,...,u_P respectively. The solution starts with the initialization of the variables from a sequential coarse time grid calculation and proceeds with an iterative procedure up to the fulfillment of a prescribed convergence criterion:

(i) Initialisation

A coarse time-grid approximation is obtained sequentially. Each processor solves the spatial field for a single time-step, Δt,

$$\begin{aligned} u_i^0 &= G_{\Delta t}(u_{i-1}^0) \\ u_0^0 &= u_0, \end{aligned} \tag{3}$$

for all processors, $i = 1, 2, .., P$.

The operator $G_{\Delta t}$ corresponds to the solution of a time step Δt. The coarse time-grid step size can also be constrained by the numerical scheme selected for the operator G. The sequential solution obtained in each processor corresponds to a coarse grain solution that requires correction, provided by the iterative scheme that follows.

(ii) Iterative Procedure

Each processor uses the previously calculated initial approximation to start an iterative procedure using the finer time-grid,

$$\begin{aligned} y_i^k &= F_{\delta t}(u_{i-1}^{k-1}) \\ u_0^k &= u_0, \end{aligned} \tag{4}$$

for $1 \leq i \leq P$ and k ≥ 1. The operator $F_{\delta t}$ denotes the parallel solution in P processors of M/P time increments from t_{i-1} to t_i.

Completed the parallel solution on the finer time-grid, the solution jumps at each t_i are calculated in parallel according to the difference between the new solution calculated on the finer time-grid and the solution on the coarse time-grid at the previous iteration,

$$S_i^k = y_i^k - u_i^{k-1}. \tag{5}$$

Finally, a new sequential solution is calculated. For $1 \leq i \leq P$ a solution is predicted using the coarse time-grid solver,

$$\tilde{u}_i^k = G_{\Delta t}(u_{i-1}^k) \tag{6}$$

corrected by the solution jumps,

$$u_i^k = \tilde{u}_i^k + \sum_{l=1}^{k} S_i^l \tag{7}$$

and communicated to the processor that is assigned to the next time-step.

3 The accuracy of the parallel-in-time numerical scheme

The accuracy of the iterative algorithm for linear parabolic differential equations was addressed by Lions, Maday, and Turinici (2001) and Bal and Maday (2002) considering the exact form of the solution on the fine time-grid but, that corresponds to a simplification in the model. The iterative scheme is ideally of order $m \times (k+1)$, being m the accuracy order of the numerical scheme considered and k the iteration number. It is important to analyze how the iterative numerical scheme behaves with some spatial and temporal discretization schemes commonly used for the solution of the Navier-Stokes equations.

For this purpose, the one-dimensional scalar transport equation

$$\frac{\partial \phi}{\partial t} + u\frac{\partial \phi}{\partial x} - \alpha\frac{\partial^2 \phi}{\partial x^2} = 0 \tag{8}$$

is applied to solve the propagating scalar pulse problem and analyze the accuracy of the parallel-in-time method. The problem consists of the one-dimensional domain from $x = 0$ to $x = 2$, through which fluid velocity is $u = 0.25$. The diffusivity was set to 10^{-3}. The initial conditions correspond to a Gaussian wave pulse with peak amplitude unity

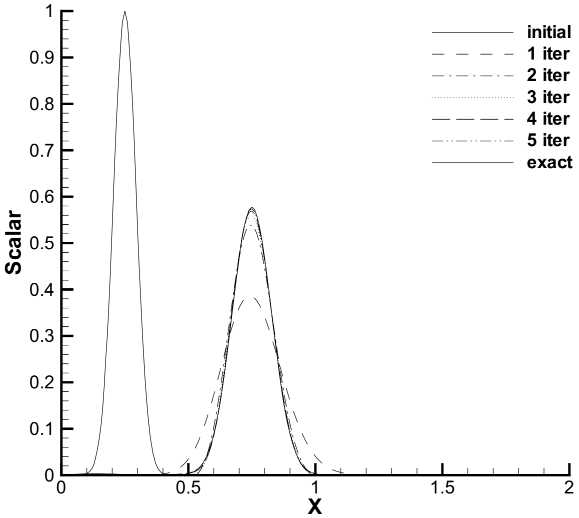

Figure 2: Initial condition ($t = 0$) and the iterative approximation to the exact solution at $t = 0$

$$\phi(x,0) = e^{-(x-u)^2/4\alpha} \tag{9}$$

considered centered at $x = 0.25$, see Fig. 2.

The time dependent solution for this problem [Yu and Heinrich (1986)],

$$\phi(x,t) = \frac{1}{\sqrt{1+t}} e^{-(x-u(1+t))^2/4\alpha(1+t)}, \tag{10}$$

allows to evaluate the error of numerical solution. The time domain considered for this purpose is T equal to 2. The numerical temporal discretization schemes considered range from first-order to fourth-order accurate. Spatial discretization of first, second and fourth-order accuracy was used together with the temporal discretization schemes to provide a stable formulation for the finite differences analog of Eq. 8. The first-order upwind spatial discretization is used together with the first-order temporal schemes (implicit and explicit Euler). For the second-order temporal schemes (Adams-Bashforth, Crank-Nicolson and three-level implicit) the second-order central differences is used for the spatial discretization and the fourth-order Runge-Kutta explicit scheme is used together with the fourth-order central differences scheme.

The numerical tests were conducted considering spatial meshes ranging from $nx = 50$ to $nx = 1600$ mesh nodes keeping CFL number constant. Therefore, the number of processors used for the calculations depends on the spatial mesh and $P = nx/2$ yields to a CFL number of 5×10^{-1} and 5×10^{-2} on the coarse and fine time-grids respectively. The parameter M, number of time-steps on the finer time-grid, was set to $10\times P$ to obtain the accurate solution on the finer time-grid required by the iterative procedure. The parallel-in-time calculations were performed in a single processor accordingly with the described algorithm simulating the required number of processors (up to 800).

Firstly, sequential solutions were performed to evaluate the accuracy of each numerical scheme on the finer time-grid. Fig. 3 shows the dependency of the L^2 norm of the error on the mesh size for several temporal schemes including also the first, second and fourth order slopes. The Figure shows that only the second and fourth-order numerical schemes are within the convergence asymptotic region of the discretization.

The dependency of the L^2 norm of the error on the spatial discretization and number of iterations performed is indicated in the Fig. 4-6 for different pairs of numerical schemes considered for the coarse and fine time-grid temporal evolution. Among the schemes tested, the fourth-order Runge-Kutta scheme is the best candidate for the fine time-grid integrator. Fig. 2 displays the initial condition ($t = 0$) and also the evolution of the solution during the iterative procedure when the first order implicit Euler scheme is used for the temporal evolution on the coarse time-grid. The evolution of the error on the spatial discretization, plotted in Fig. 4, shows that the order of accuracy increases with the number of iterations but does not reach the formal convergence order $(k+1)$ because the first-order scheme is not in the convergence asymptotic region.

It is possible to detect the same behavior for the second-order accurate Crank-Nicolson scheme for the coarse time-grid solution. However, after the second/third iteration, depending on the discretization, no further improvement on the accuracy of the solution is obtained because the accuracy of the fine time-grid solution is already acquired, see Fig. 5.

The second-order accurate Adams-Bashforth explicit scheme is also a good candidate for the numerical scheme to use for the fine time-grid solution because, although only second-order accurate, is computationally less expensive than the Runge-Kutta scheme. For the problem under consideration, the convergence is achieved after four iterations when the Adams-Bashforth is used together with the implicit Euler scheme on the coarse time-grid, see Fig. 6.

The selection of the numerical schemes for the temporal evolution on each time-grid, besides the stability constrains, should be made considering that the numerical scheme for the finer time-grid evolution should provide higher order of convergence than the one used in the coarse time-grid. When using the fourth-order accurate numerical scheme on the fine time-grid, the convergence of the iterative parallel-in-time method requires fewer iterations using a second-order accurate scheme for the coarse time-grid than an unconditionally stable first-order scheme. The speed-up obtained with the parallel-in-time method is strongly dependent on the number of iterations required to meet the convergence criterion. The number of iterations to acquire the accuracy of the fine-grid solution is obviously dependent of the nature of numerical schemes applied. Considering the general theory related with the convergence of numerical schemes, one can approximate the number of iterations required for convergence by the ratio between the order of accuracy of the numerical schemes used for the coarse and fine time-grids solution. This prediction is clearly verified when using the second-order accurate Crank-Nicolson and the fourth-order Runge-Kutta schemes for the coarse and fine time-grid solutions respectively. When using the formal first-order implicit Euler scheme for the coarse time-grid solution one must consider that the accuracy order is only 0.61 evaluated in the interval between $\Delta x = 0.00125$ and $\Delta x = 0.005$ on the initial approximation. There-

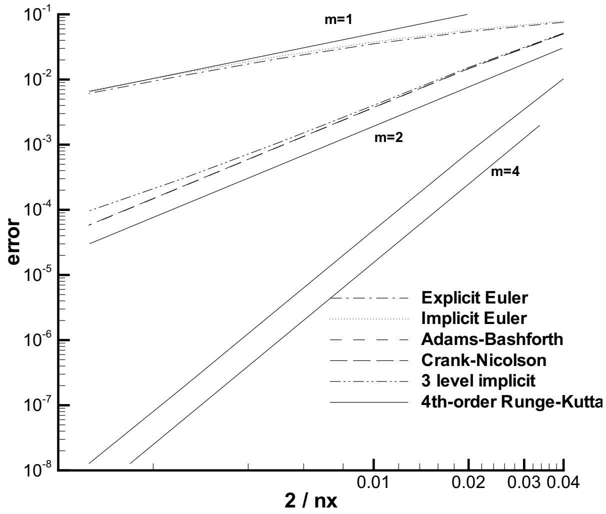

Figure 3: Dependency of sequential solution on the fine time-grid L^2 error norm on the mesh resolution.

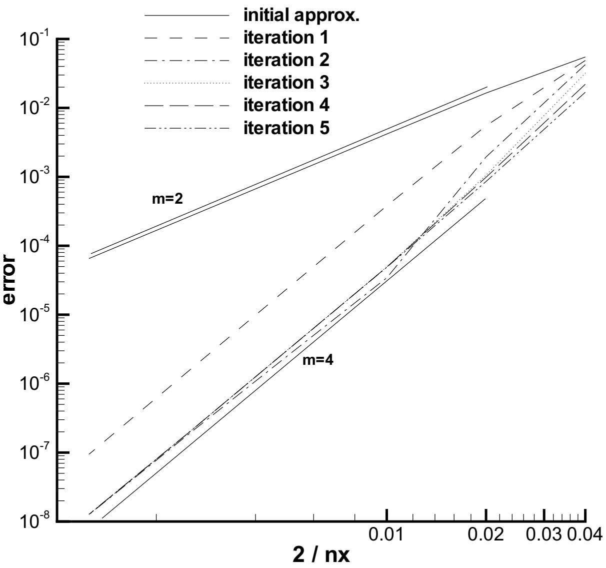

Figure 5: Error dependency on the spatial discretization and number of iterations using the Crank-Nicolson and fourth-order Runge-Kutta schemes on the coarse and fine time-grids respectively.

fore, the prediction of the number of iterations required to achieve the accuracy of the Runge-Kutta or the Adams-Bashforth schemes fine time-grid solution is 7 and 4, respectively.

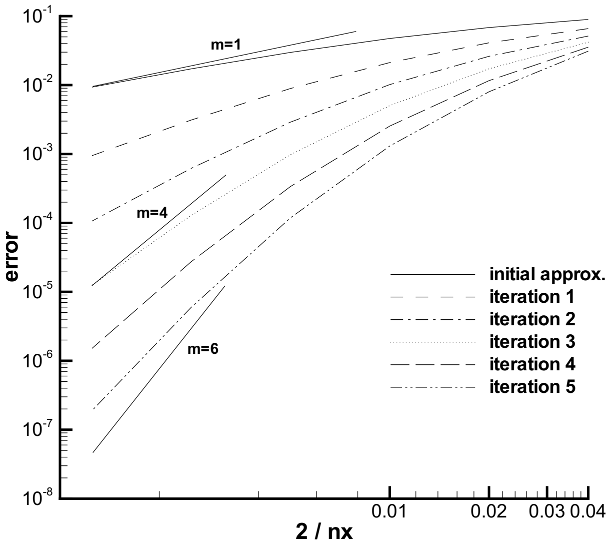

Figure 4: Error dependency on the spatial discretization and number of iterations using the implicit Euler and fourth-order Runge-Kutta schemes on the coarse and fine time-grids respectively.

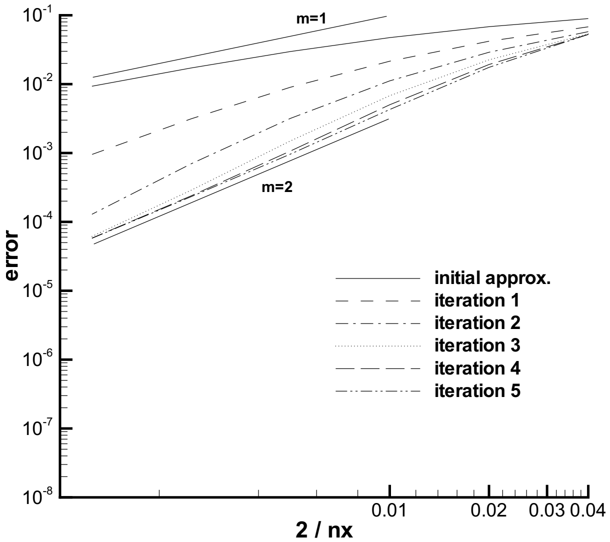

Figure 6: Error dependency on the spatial discretization and number of iterations using the implicit Euler and Adams-Bashforth schemes on the coarse and fine time-grids respectively.

4 The performance model

Besides the number of iterations required for convergence, the parallel efficiency of the method is also strongly dependent on the time spent by the communication tasks required by the algorithm. The following is an attempt to derive theoretically the parallel speed-up of the method presently used, even with some simplifying assumptions, in order to evaluate the potential application of the technique for the unsteady incompressible Navier-Stokes equations. The comparison between the performance prediction and the speed-up effectively achieved with a PC-cluster computation allows to verify how relevant is the time spent on communication tasks for the efficiency of the parallel-in-time method.

Fig. 1 displays two iterations of the cycle. The computing time required to perform the first iteration is denoted by T_1. One should note that after the first iteration the solution at time t_1 corresponds to the final solution at this time level. More generally, at iteration k the solution at time t_k does not need further iterations because the final solution is achieved. Considering the overlap between sequential and parallel tasks indicated in Fig. 1 and neglecting the communication time, the total computing time of a parallel-in-time solution using P processors, Γ_P, can be predicted by

$$\Gamma_P = \Gamma_{seq} + k\frac{\Gamma_{seq}+\Gamma_1}{P} \tag{11}$$

where Γ_{seq} denotes the computing time of the sequential solution on the coarse time-grid, Γ_1 the computing time on a single processor and k is the number of iterations prescribed. Parallel speed-up of a computation, defined as

$$S = \frac{\Gamma_1}{\Gamma_P} \tag{12}$$

can then be predicted by

$$S = \frac{\Gamma_1}{\Gamma_{seq} + k\frac{\Gamma_{seq}+\Gamma_1}{P}} \tag{13}$$

Considering the same computing time for one time step on the fine and coarse time-grid, the maximum expected speed-up that can be achieved is approximated by

$$S = \frac{M}{P + k\left(1+\frac{M}{P}\right)} \tag{14}$$

where M is the number of time-steps considered on finer time-grid. Fig. 7 shows the predicted speed-up of a parallel solution, Eq. 14, as a function of the number of processors and the ratio between the coarse and the fine time-grid step sizes, when two iterations on the parallel-in-time algorithm are required to meet the convergence criterion. The Figure shows that for small time-step ratios no substantial speed-up is achieved. However, important parallel speed-up can be predicted for high time-step ratios. Similar conclusions could be derived considering other number of iterations, k. For high time-step ratios, the speed-up always increases with the number of processors and is limited by P/k.

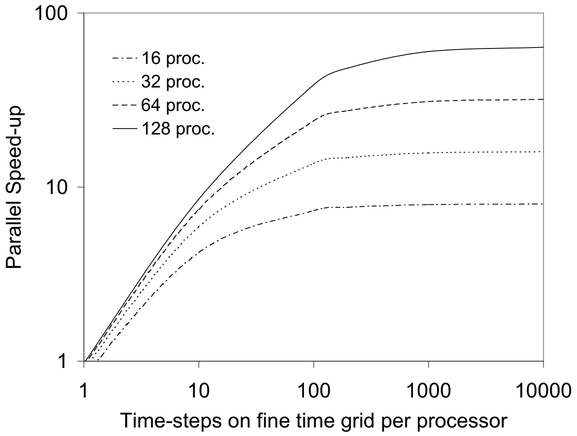

Figure 7: Parallel-in-time speed-up prediction for two iterations.

The behavior of parallel-in-time processing is rather different from the space domain decomposition when, for a fixed spatial dimension of the problem, speed-up has a limit imposed by the calculation/communication time ratio and no further speed-up improvement can be achieved by increasing the number of processors involved. The penalty on the parallel-in-time method is inherent to the algorithm and consequently, the efficiency comparison of the present method with the classical spatial domain decomposition method is adverse in most cases. However, despite the low efficiency obtained, important computing time reductions could be accomplished if a large amount of processors are available and a few iterations are required to converge the solution.

The parallel-in-time method is applied for the numerical simulation of the laminar flow past a two-dimensional square cylinder between parallel walls at the supercritical Reynolds number of 500 to evaluate the proposed performance model. Trindade and Pereira (2004) presented parallel-in-time simulation of similar confined flows, although with different numerical schemes, illustrating the application of the parallel-in-time method to a demanding unsteady flow problem. Flow around bluff bodies is characterized by the onset of periodic oscillations, the von Karman vortex street, after a critical Reynolds number, consisting of a difficult non-linear case for the iterative parallel-in-time algorithm.

Equations 1 and 2 can be discretized by very many ways, see e.g. Ferziger and Peric (1997). Although not convenient for the accuracy of the solution, the operators $G_{\Delta t}$ and $F_{\delta t}$ uses the same spatial and temporal numerical schemes to approximate the assumption of equal computing time for one time step on the fine and coarse time-grids.

For the purpose of this work, it is sufficient to use a deferred correction scheme that blends second-order central and upwind differences on a finite volume staggered uniform spatial grid. This blended discretization scheme, with coefficient 0.2 for the upwind contribution, removes the oscillations produced by the central differences schemes on the mesh Reynolds numbers considered. The temporal discretization was performed by the implicit Crank-Nicolson scheme yielding to Eq. 2 to appear as:

$$\frac{u^{n+1}-u^n}{\Delta t}\Omega+\left[\frac{1}{2}\sum_{i\in S}C_i^n+\frac{1}{2}\sum_{i\in S}C_i^{n+1}\right] -\left[\frac{1}{2}\sum_{i\in S}D_i^n+\frac{1}{2}\sum_{i\in S}D_i^{n+1}\right]=-G^n \quad (15)$$

where C_i and D_i stands for the convective and diffusive fluxes evaluated at the time level n and $n+1$ and G represents the pressure source term. At each time-step calculation, the SIMPLE method Patankar (1980) is used to correct the velocity and pressure fields enforcing a divergence free velocity field. One should note that the basic numerical method considered is one among many others of different order of accuracy that could be used to validate the proposed performance model.

The square cylinder, of width unity, is situated symmetrically between two walls a distance H = 4 apart. Blockage ratio is therefore $1/4$ and the channel length is equal to 24. A uniform flow was prescribed at the inlet. At the outlet, a convective boundary condition was used for velocity components. No-slip conditions were prescribed at body surfaces and at upper and lower boundaries. The flow is impulsively started from a initial rest condition and a constant time-step on the coarse grid, Δt, is used. The calculation of the described numerical experiment was performed on a PC-cluster with 16 nodes, each one with one Pentium IV 2.4 GHz processor and 512 Mb Ram. A 100 Mbps ethernet switch is used for the node connection. Spatial domain is discretized on a uniform 151×26 nodes grid. Stability constraints related with the numerical method imposed a coarse grid time-step size equal to $\Delta t = 0.1$. As the number of processors available is insufficient to perform a parallel-in-time calculation corresponding to the simulation time of several shedding periods, that would require 2000 processors for a time interval T equal to 200, time-blocks were considered. In this way, time-blocks with size equal to $P\times\Delta t$ are solved sequentially. The fine time-grid step size is equal to $\delta t = 0.01$.

The temporal discretization scheme used in the coarse time-grid was also used in the finer time-grid. Many other options could be used, see e.g. the schemes listed in Fig. 3. The reason why this option was taken relies on the assumption used in Eq. 14 in order to get close computing times for each time-step on both time-grids. The purpose of present work is to evaluate the parallel efficiency of the method and compare with the performance model rather than to investigate in detail the physics of self-sustained wake oscillations that would require either better spatial resolution or high-order spatial schemes. As a few iterations are required for convergence, no convergence criterion was applied and the number of iterations performed was prescribed (two and three iterations). Fig. 8 shows the vorticity contours after the first and the second iteration on the parallel-in-

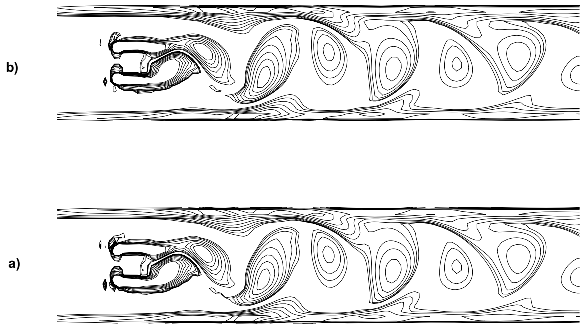

Figure 8: Vorticity contours after first (a) and second (b) iteration.

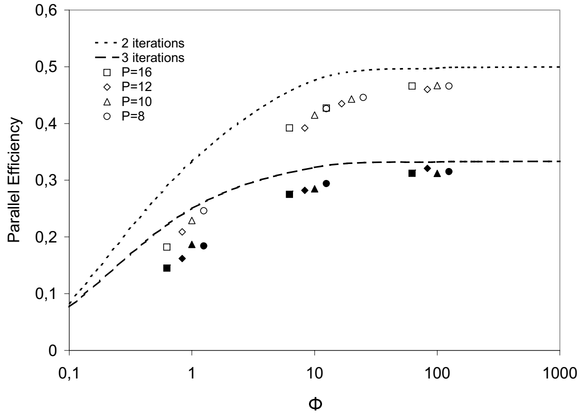

Figure 9: Comparison between predicted (lines) and verified (symbols, filled symbols are related to 3 iterations) efficiency of the parallel-in-time method for 2 and 3 iterations prescribed ($\Phi = M/P^2$).

time algorithm after the establishment of the periodic flow. Very small differences are detectable on the vorticity contours of those flow fields. The flow field after the third iteration is virtually equal to the one obtained after the second iteration.

Ten time-blocks, consisting each one of P time-steps, were calculated using different number of processors (1, 8, 10, 12, and 16) and different coarse to fine time-grid step size ratios (10, 100, 200 and 1000). In order to avoid the influence of the initial time steps on an impulsive start from rest, a flow field solution after the establishment of the periodic flow was used as the initial condition to start these calculations. The sampling frequency, $\Delta t = 0.1$, was kept unchanged in all the cases. The use of a prescribed number of time-blocks on the comparison instead of a prescribed time interval is necessary to allow the use of the above mentioned cluster dimensions maintaining the sampling frequency. Two and three iterations on the parallel-in-time algorithm were considered. Fig. 9 shows the comparison between the verified parallel efficiency of the method and predictions based on the performance model, Eq. 14. The predicted efficiency that depends on M, P and k was plotted only as a function of $\Phi = M/P^2$ for each number of iterations prescribed on the parallel-in-time algorithm. Fig. 9 shows good agreement between predicted and verified parallel efficiency. Major deviations are verified for small values of Φ when the computing time of the initial sequential coarse time-grid calculation gives a more important contribution to the total computing time. For large values of Φ one should conclude that the the communication overhead does not prevent the application of the present method for long-term fluid flow problems.

5 Conclusions

A parallel-in-time method, based on temporal domain decomposition, was applied for the solution of the unsteady incompressible Navier-Stokes equations. To evaluate the potential efficiency of the present method to fluid flow simulations, the convergence of this iterative method was analyzed considering some spatial and temporal discretization schemes commonly used for that purpose. The one-dimensional scalar transport equation allowed some conclusions concerning the accuracy and convergence of the iterative method. Another important issue related with the performance of the parallel-in-time method is the time spent by the communication tasks required by the algorithm. A simplified performance model for the parallel-in-time method was proposed. The flow past a two-dimensional square cylinder between parallel walls for Reynolds number equal to 500 was selected to analyze through numerical experiments the influence of the communication time on the parallel efficiency of the method. The following conclusions could be derived:

(i) The accuracy of the parallel-in-time solution increases with the number of iterations accordingly with the order of accuracy of the numerical schemes considered for the coarse and fine time-grids. The number of iterations required for convergence can be estimated by the ratio between those orders of accuracy.

(ii) The agreement between the verified parallel efficiency in the numerical experiments and the one given by the theoretical model indicates that the communication overhead does not impose a critical limitation on the application of the present methodology for the unsteady incompressible Navier-Stokes equations.

(iii) For large ratios between the coarse and fine time-step sizes, substantial computing time reduction can be expected.

(iv) The results suggest that the parallel-in-time methodology is promising when the temporal scale of the problem under consideration is large and a large amount of processors is available.

Some issues, like the optimal choice of the numerical methods for coarse and fine time-grids, need further research to increase the robustness and efficiency of the method.

References

Baffico, L.; Bernard, S.; Maday, Y.; Turinici, G.; Zerah, G. (2002): Parallel-in-time molecular-dynamics simulations. *Int. J. Quant. Chem.*, vol. E66, pp. 057701.

Bal, G.; Maday, Y. (2002): A 'parareal' time discretization for non-linear PDE's with application to the pricing of an american put. In Pavarino, L.; Toselli, A.(Eds): *Proceedings of the Workshop on Domain Decomposition, Zurich, Switzerland*, volume **23** of *Lecture Notes in Computer Science and Engineering Series*, Berlin. Springer.

Farhat, C.; Chandesris, M. (2003): Time-decomposed parallel time-integrators: theory and feasibility studies for fluid, structure, and fluid-struture applications. *Int. J. Numer. Methods Engng.*, vol. 58, pp. 1397–1434.

Ferziger, J.; Peric, M. (1997): *Computational methods for fluid dynamics*. Springer-Verlag.

Grimaldi, A.; Pascazio, G.; Napolitano, M. (2006): A parallel multi-block method for the unsteady vorticity-velocity equations. *CMES: Computer Modeling in Engineering & Sciences*, vol. 14, no. 1, pp. 45–56.

Lions, J.; Maday, Y.; Turinici, G. (2001): Resolution d'EDP par un schema en temps 'pararéel'. *Comptes Rendus de l Academie des Sciences, Paris, Séries I, Mathematique*, vol. 332, pp. 661–668.

Maday, Y.; Turinici, G. (2003): Parallel in time algorithms for quantum control: the parareal time discretization scheme. *Int. J. Quant. Chem.*, vol. 93, no. 3, pp. 223–228.

Nicolás, A.; Bermúdez, B. (2004): 2D incompressible viscous flows at moderate and high Reynolds numbers. *CMES: Computer Modeling in Engineering & Sciences*, vol. 6, no. 5, pp. 441–451.

Patankar, S. (1980): *Numerical heat transfer and fluid flow*. Hemisphere Publishing Corp.

Shu, C.; Ding, H.; Yeo, K. (2005): Computation of incompressible Navier-Stokes equations by local RBF-based differential quadrature methods. *CMES: Computer Modeling in Engineering & Sciences*, vol. 7, no. 2, pp. 195–205.

Trindade, J.; Pereira, J. (2004): Parallel-in-time simulation of the unsteady Navier-Stokes equations for incompressible flow. *Int. J. Numer. Methods Fluids*, vol. 45, pp. 1123–1136.

Trindade, J.; Pereira, J. (2006): Parallel-in-time simulation of two-dimensional unsteady incompressible laminar flows. *Num. Heat Transf. B, Fundamentals*, vol. 50, pp. 25–40.

Yu, C.; Heinrich, J. (1986): Petrov-Galerkin methods for the time-dependent convective transport equation. *Int. J. Numer. Methods Engng.*, vol. 23, pp. 883–901.