ParCFD2023
34^{th} International Conference on Parallel Computational Fluid Dynamics
May 29-31 2023, Cuenca Ecuador

STRATEGIES FOR INCREASING THE ARITHMETIC INTENSITY ON ENSEMBLE AVERAGED PARALLEL-IN-TIME SIMULATIONS

Josep PLANA-RIU*, F.Xavier TRIAS*, Àdel ALSALTI-BALDELLOU*,†, Asensio OLIVA*

*Heat and Mass Transfer Technological Centre
Technical University of Catalonia, ESEIAAT
Carrer de Colom 11, 08222 Terrassa (Barcelona), Spain
e-mail: josep.plana.riu@upc.edu

† Termo Fluids S.L.
Carrer de Magí Colet 8, 08204 Sabadell (Barcelona), Spain

Key words: Computational Fluid Dynamics, Parallel-in-time, Ensemble Averaging, Sparse matrix-matrix product, Sparse matrix-vector product, Arithmetic intensity

Summary. Extracting flow statistics from flow simulations requires generally long integration intervals in which a good deal of the time is spent time-averaging the results. In order to avoid this, ensemble averaging can be applied to reduce the overall simulation time by simulating multiple statistically independent flow states for a smaller period of time and later averaging among them. In order to optimize the process, a cross-platform portable parallel-in-time method is presented in which the overall simulation is solved making use of sparse matrix-matrix products (SpMM), a high arithmetic intensity counterpart for sparse matrix-vector products, so that the efficiency of the simulation is increased.

1 INTRODUCTION

Accurate simulations of the incompressible Navier-Stokes equations with high Reynolds numbers has been one of the most common applications for high performance computing (HPC) systems, since the large grids required to capture all the resolved scales lead to parallelization in order to reduce the time spent in each simulation. Even though the computational power available is increasing constantly, the possibility of performing a DNS (or even LES) for most of real life applications is still rare, extremely time consuming, and computationally expensive.

Nonetheless, the parallelization is generally applied to space, since time is essentially a sequential process and thus, time-parallelization would seem difficult to apply given that time is usually advanced explicitly, being the most common options the second-order Adams-Bashforth (AB2) or three- or four-stage Runge-Kutta (RK3,RK4) schemes. However, Falgout et al. [1] proposed a parallel-in-time method for parabolic problems, in which a limited set of time-discretizations were iteratively solved in a multigrid-like approach,

the Multigrid Reduction In Time (MGRIT). By applying it to the incompressible Navier-Stokes equations, Christopher et al. [2] obtained a 7-fold speed up on 1024 processors when compared to the time-sequential simulation, for the Stokes second problem.

Other parallel-in-time techniques that have been applied to the incompressible Navier-Stokes equations are those presented by Krasnopolsky [3] in which multiple simulations are run simultaneously in time, with CPU parallelization, by exploiting the possibility of solving simultaneously multiple sparse matrix-vector products (SpMV) in the so-called generalized sparse matrix-vector products (GSpMV) or sparse matrix-matrix products (SpMM). By doing so, a memory-bound operation such as the SpMV which generally does not exceeed a small percentage of the peak prformance [4] has its arithmetic intensity increased so that the computational efficiency improves. After doing so, ensemble averaging techniques such as the presented by Marakashvili et al. [5], in which the flow statistics can be obtained from multiple shorter simulations, or flow states, rather than from a long time-averaged simulation, are applied. With this technique, Krasnopolsky [3] obtained speed-ups between 1.5- to 2-fold in the test cases, which coincide with the theoretical predictions.

2 PARALLEL-IN-TIME ENSEMBLE AVERAGING

The methodology for parallel-in-time ensemble averaging is based on the works of Krasnopolsky [3]. The technique is based on the concept of solving multiple (d) flow states simultaneously, in order to time-average the results within each of the flow states and then perform an average among the flow states in order to obtain the flow statistics [5]. In order to exploit the low arithmetic intensity (AI) of SpMV, these multiple flow states are solved in the same device and the resolution of the Poisson equation is performed in parallel for all the flow states, in the so-called sparse matrix-matrix product (SpMM), which corresponds to a product of a sparse matrix with a dense matrix with n columns; while the rest of the SpMV operations are performed normally.

The arithmetic intensity of the SpMM product with n columns can be stated as in (1), by assuming ideal temporal locality, where A is the sparse matrix, $nnz(A), m, n$ are the number of non-zero entries, rows and columns in the matrix, respectively; and d is the number of flow states. Hence, the upper bound of the theoretical speed-up, $P_{d,ub}$ for an SpMM compared to its equivalent SpMV with the same sparse matrix is set as $AI_{\text{SpMM}}/AI_{\text{SpMV}}$, which can be rewritten as in (2), while the lower-bound, $P_{d,lb}$ can be processed exactly as for the upper-bound, but considering zero temporal locality (and thus replacing $nnz(A)$ by n), leading to the speed-ups from Figure 1 according to the number of right-hand side (RHS) vectors.

$$AI_{\text{SpMM}}(d) = \frac{(2nnz(A)+1)d}{12nnz(A)+4(m+1)+8(m+n+1)d}, \quad (1) \qquad P_{d,ub} = \frac{AI_{\text{SpMM}}(d)}{AI_{\text{SpMM}}(1)}, \quad (2)$$

Regarding the parallel implementation of the methodology, the mesh used for all the concurrent flow states will be distributed across multiple hybrid cluster nodes. Hence,

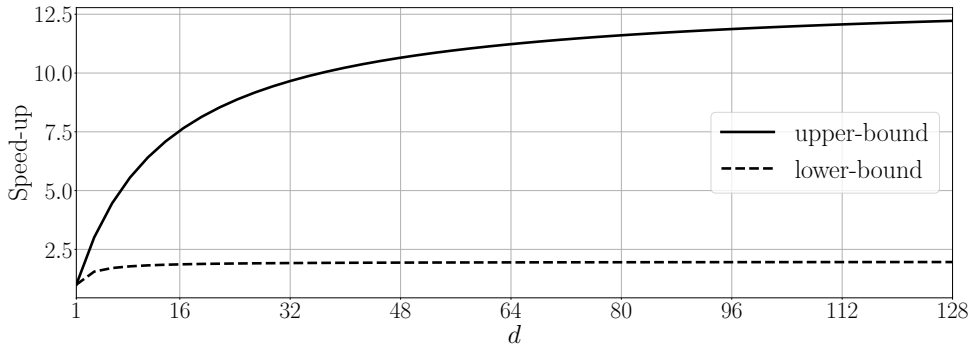

Figure 1: Theoretical speed-up bounds for a sparse matrix A with $nnz(A)/m = 17$.

each of these splittings will be further split into either CPUs or GPUs. The usage of hybrid nodes, together with its algebra-based framework, will make the cross-platform portability natural, and thus the incorporation of external libraries (e.g. cuSPARSE [6]) in the developed methodology will be simplified.

3 APPLICATION TO CFD SIMULATIONS

The governing equations for a DNS method are the incompressible Navier-Stokes equations, which semi-discretized following the notation of [7] read as follows,

$$M\mathbf{u}_s = \mathbf{0}_c, \qquad (3) \qquad\qquad \Omega\frac{d\mathbf{u}_c}{dt} + C(\mathbf{u}_s)\mathbf{u}_c - D\mathbf{u}_c + \Omega G_c\mathbf{p}_c = \mathbf{0}_c, \qquad (4)$$

where M is the face-to-cell divergence operator, Ω_c is a diagonal matrix containing the cell volumes so that $\Omega = I_3 \otimes \Omega_c$, C_c is the cell-to-cell convective operator so that $C = I_3 \otimes C_c$, D_c is the cell-to-cell diffusive operator so that $D = I_3 \otimes D_c$, G_c is the cell-to-cell gradient operator, $\mathbf{u}_s$ is the velocity field defined at the faces, and I_3 is the identity matrix of size 3.

Eq. (4) will be integrated in time with a third-order Runge-Kutta scheme [8] within the framework of the projection method [9].

Hence, changing from SpMV to SpMM can be applied to each of the SpMV operations that appear in an iteration of a third-order Runge-Kutta scheme applied to a projection method (around 30 SpMV apart from the resolution of the three Poisson equations). This leaves room for improvement in the method presented by Krasnopolsky [3], in which this increase in the arithmetic intensity was applied only in the resolution of the system of linear algebraic equations (SLAE), while the presented work intends to increase the compute intensity in the totality of the iteration.

Note that the numerical scheme used for the spatial discretization will determine the sparsity pattern as well as the density of the operator matrices $C(\mathbf{u}_s), D, G$. According to the definition of the speed-up upper-bound (2), which increases with increasing densities of

the sparse matrix ($nnz(A)/m$), it can be seen that the effect of replacing SpMV by SpMM will be enhanced by using higher-order numerical schemes for the spatial discretization.

Hence, the parallel-in-time ensemble averaging methodology will be tested in canonical cases in order to compare the speed-ups obtained with the fully-SpMM case with the ones obtained with SLAE-SpMM case [3].

Acknowledgements. J.P-R., F.X.T., A.A-B. and A.O. are supported by the *Ministerio de Economía y Competitividad*, Spain, RETOtwin project (PDC2021-120970-I00). J.P-R. is also supported by the Catalan Agency for Management of University and Research Grants (AGAUR), and A. A-B. is also supported by the predoctoral grants DIN2018-010061 and 2019-DI-90, by MCIN/AEI/10.13039/501100011033 and AGAUR. Calculations are being performed on the MareNostrum 4 supercomputer at the BSC. The authors thankfully acknowledge these institutions.

References

[1] R. D. Falgout, S. Friedhoff, T. V. Kolev, S. P. MacLachlan, and J. B. Schroder, "Parallel time integration with multigrid," *SIAM Journal on Scientific Computing*, vol. 36, pp. C635–C661, 6 2014.

[2] J. Christopher, X. Gao, S. Guzik, R. D. Falgout, and J. B. Schroder, "Parallel in time for a fully space-time adaptive mesh refinement algorithm," in *AIAA SciTech Forum and Exposition*, 2019.

[3] B. I. Krasnopolsky, "An approach for accelerating incompressible turbulent flow simulations based on simultaneous modelling of multiple ensembles," *Computer Physics Communications*, vol. 229, pp. 8–19, 2018-08.

[4] W. Gropp, D. Kaushik, D. Keyes, and B. Smith, "Towards realistic performance bounds for implicit CFD codes.," in *International Conference on Parallel Computational Fluid Dynamics*, 1999.

[5] V. Makarashvili, E. Merzari, A. Obabko, A. Siegel, and P. Fischer, "A performance analysis of ensemble averaging for high fidelity turbulence simulations at the strong scaling limit," *Computer Physics Communications*, vol. 219, pp. 236–245, 2017.

[6] "cuSPARSE: The API reference guide for cuSPARSE, the CUDA sparse matrix library," NVIDIA Corporation, 2020.

[7] F. X. Trias, O. Lehmkuhl, A. Oliva, C. D. Pérez-Segarra, and R. W. Verstappen, "Symmetry-preserving discretization of Navier-Stokes equations on collocated unstructured grids," *Journal of Computational Physics*, vol. 258, pp. 246–267, 2014-02.

[8] B. Sanderse and B. Koren, "Accuracy analysis of explicit Runge-Kutta methods applied to the incompressible Navier-Stokes equations," *Journal of Computational Physics*, vol. 231, pp. 3041–3063, 8 2012-04.

[9] A. J. Chorin, "Numerical solution of the Navier-Stokes equations," *Mathematics of Computation*, vol. 22, pp. 745–762, 104 1968.