A MPI-CUDA PARALLEL SOLVER FOR 3D FLOWS ON UNSTRUCTURED FINITE VOLUME MESHES

Miguel Uh Zapata[*] **and Francisco J. Hernández-López**[*]

[*] CONACYT - Centro de Investigación en Matemáticas A.C, Unidad Mérida
97302 Mérida, Yucatán, MEXICO
e-mail: angeluh@cimat.mx, web page: http://www.cimat.mx/∼angeluh
e-mail: fcoj23@cimat.mx, web page: http://www.cimat.mx/∼fcoj23

Key words: 3D Navier-Stokes equation, Unstructured mesh, MPI, GPU computing, CUDA Fortran, Multi-color SOR method.

Abstract. In this paper, we present a parallelization technique developed to solve an unstructured-grid based Navier-Stokes equations on arbitrary three-dimensional geometries using a hybrid MPI-CUDA parallel solver. The discretization is based on a second-order finite-volume technique on prisms elements, consisting of triangular grids in the horizontal direction and bounded by an irregular free surface and bottom in the vertical direction. The overall parallelization is designed by a block domain decomposition technique using MPI and the resulting pressure linear systems is solved using a Multi-color Successive Over-Relaxation method on GPU. Numerical experiments are conducted to test and compare the performance of the proposed parallel technique.

1 INTRODUCTION

Although typically the CPU (Central Processing Unit) can be used to fluid dynamics simulations with unstructured grids [1], current trent is to perform parallel computations using GPU (Graphics Processing Unit) due to its increasing computational power and low cost. In order to achieve a good parallel performance, we must be careful to consider proper techniques that could be applied to real cases as shown in Fig. 1.

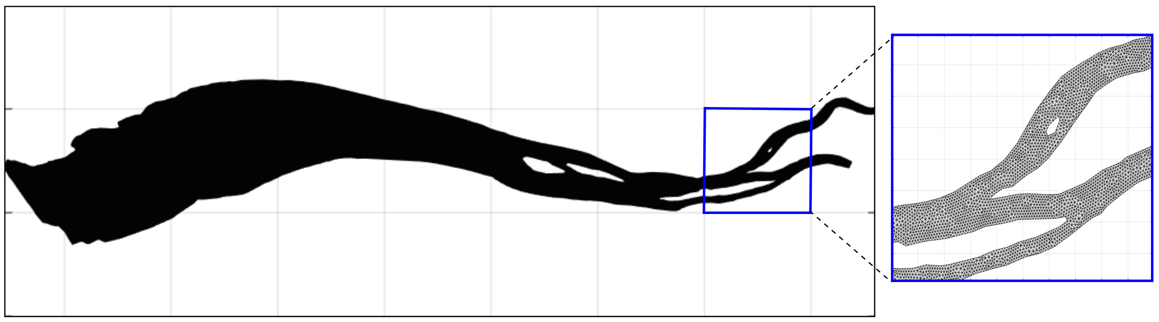

Figure 1: Horizontal domain and unstructured mesh for an estuary example (Gironde, France).

There are several techniques used to parallel such as Open Multi-Processing (OpenMP), Message passing Interface (MPI), Compute Unified Device Architecture (CUDA), etc. However, there are few works using parallel methods with a finite volume approach on 3D irregular domains [1, 2, 3]. The general discretization of the problem produces a large amount of data; thus a domain decomposition approach with a distributed memory is suitable for this problem. However, the pressure Poisson equation still represents both the most time consuming part of the parallel code and the main source of performance bottlenecks. Motivated by these facts, a MPI algorithm is presented for the finite volume solution of the 3D Navier-Stokes equation and a CUDA parallel solver is used to improve the pressure calculation time. The proposed parallel method uses the Multi-color SOR (MSOR) method to solve the linear systems due to its stability with a large matrix size and its simplicity to parallelize without many changes of the program. The authors are not aware of publications with MPI-CUDA implementations of similar finite volume discretizations.

2 METHODOLOGY

The SOR method is difficult to parallelize, particularly on GPUs due to the existence of data dependence [4]. The typical way to process in a GPU consist in to transfer the input data from CPU memory to GPU memory, next the data are processed into a kernel function, then the output data are transferred from GPU memory to CPU memory. The CUDA programming model can run thousands of threads that are grouped into blocks. For each GPU kernel, we are allowed to configure several aspects of the grid, including the number of threads per block as well as the number of blocks. For this particular problem, elements are processed in rectangular blocks of data assigned to threads as sketch in Fig. 2. This configurations was chosen to take advantage of the element distribution of the problem and easy code implementation.

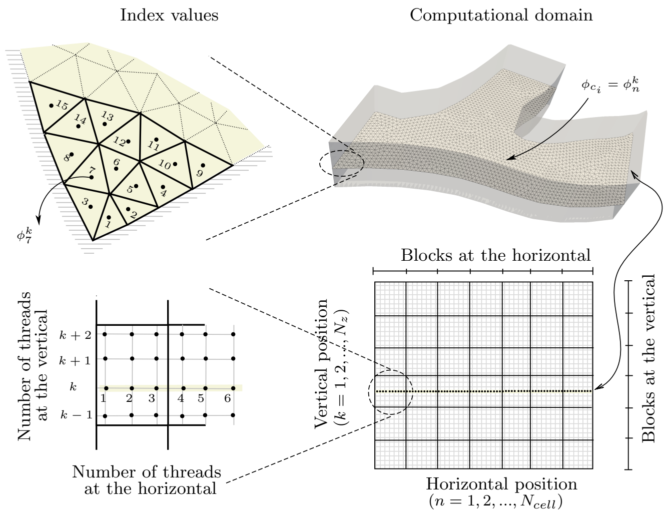
Index values
Computational domain
$\phi_{c_i} = \phi_n^k$
ϕ_7^k
Blocks at the horizontal
Number of threads at the vertical
$k+2$
$k+1$
k
$k-1$
Vertical position ($k = 1, 2, ..., N_z$)
Blocks at the vertical
Number of threads at the horizontal
Horizontal position ($n = 1, 2, ..., N_{cell}$)

Figure 2: Topology distribution of the blocks and threats in the CUDA parallelization.

An essential part to maximize the performance of our parallel implementation is the GPU memory management. Kernel functions can read and write in the GPU device global memory using a controller integrated into the GPU. Each thread block has a limited shared memory space; however, it is faster than accesses to global memory. The texture memory is a read-only cache, basically is a reference to a CUDA array and contains information on how the array should be interpreted. On the other hand, the unified memory defines a managed memory space in which all CPU and GPU processors see a single coherent memory image with a common address space. In this paper, we analyze three code versions according to the type of memory used to parallelize the MSOR method. In the first version (CUDA D), we only use the device global memory, in the second version (CUDA DT), we use device global and texture memory, and in the third version (CUDA DTM), we use managed memory in addition to device global and texture memory.

3 NUMERICAL RESULTS

The GPU performance of the proposed method is initially investigated in terms of the total time simulation and speedup of the MSOR method. For validation and comparison purposes we applied our parallel solvers to the solution of the pressure Poisson equation with a known analytical solution. It is given by $p(x, y, z) = \sin(\pi x)\sin(\pi y)\sin(\pi z)$. The numerical simulations were performed using an irregular domain with different mesh resolutions in the horizontal and vertical direction as shown in Fig. 3. Mesh 1, 2 and 3 are approximately 1, 20 and 100 thousands of cell-centered points, respectively. The finest mesh can reach a total of 12 millions elements.

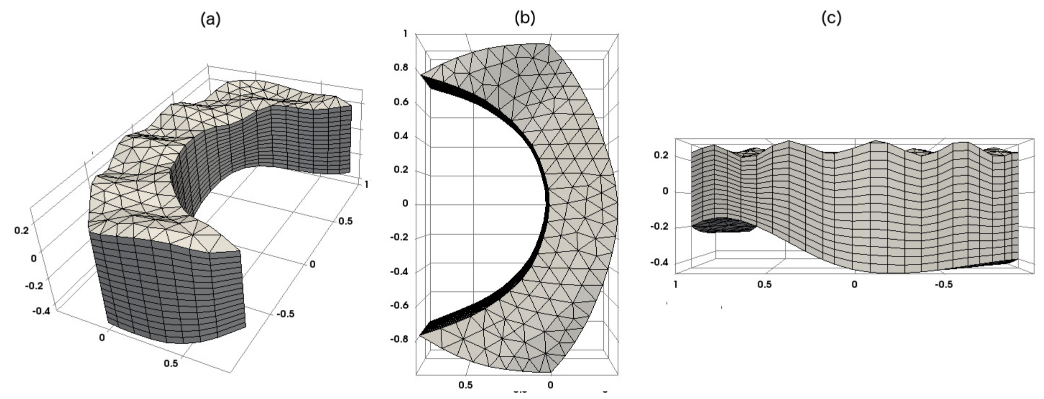

Figure 3: Irregular domain using different view points.

Table 1 display the partial results explaining the relationship between the problem size and parallel efficiency for the MSOR method. We notice that the problem size is a factor that influences parallel performance. If more mesh points are used, then the Speedup increases. The results of this example were obtained using a standard twenty-core 2.1 GHz Intel Xeon machine with a Tesla K20.

Table 1: Performance using different grid resolutions.

		Sequential	CUDA D		CUDA DT		CUDA DTM	
Mesh	N_z	Time	Time	Speedup	Time	Speedup	Time	Speedup
1	32	1.02	0.18	5.67 x	0.17	6.00 x	0.18	5.67 x
1	64	6.01	0.77	7.80 x	0.73	8.23 x	0.74	8.12 x
1	128	40.10	5.40	7.42 x	6.11	6.56 x	5.10	7.86 x
2	32	395.74	25.72	15.38 x	23.24	17.02 x	21.59	18.32 x
2	64	944.98	54.86	17.22 x	46.39	20.37 x	45.54	20.75 x
2	128	2,615.94	156.99	16.66 x	140.10	18.67 x	138.90	18.83 x
3	32	6,590.88	382.14	17.24 x	324.90	20.28 x	315.28	20.90 x
3	64	14,521.09	778.38	18.65 x	650.45	22.32 x	636.39	22.81 x
3	128	33,800.46	1775.03	19.04 x	1582.31	21.36 x	1566.36	21.57 x

4 CONCLUSIONS

In this paper we present results for a parallel unstructured finite volume method for the 3D Navier-Stokes equations using a block domain decomposition and a Multi-color ordering technique on CUDA. Such method provide an attractive way to obtain fast simulations for 3D irregular domains. In addition, the parallel algorithm is straightforward and good speedup could be obtained with several configurations. The initial results were tested in different grids and different domain shapes. Results are extremely promising, although more numerical experimentation is still need. Current work involves a parallel code to obtain 3D flow simulations on real irregular domains using the proposed finite volume technique. Further work will involve optimization techniques in the code to obtain faster simulations.

REFERENCES

[1] Uh Zapata, M., Van Bang, D. P., and Nguyen, K. D. (2016). Parallel SOR methods with a parabolic-diffusion acceleration technique for solving an unstructured-grid Poisson equation on 3D arbitrary geometries. International Journal of Computational Fluid Dynamics, 30(5): 370-385.

[2] Y. Sato, T. Hino, K. Ohashi, (2013). Parallelization of an unstructured NavierStokes solver using a multi-color ordering method for OpenMP. Computers & Fluids, 88: 496-509.

[3] M. Cheng, G. Wang, H. Hameed Mian, (2014). Reordering of hybrid unstructured grids for an implicit Navier-Stokes solver based on OpenMP parallelization, Computers & Fluids .

[4] Konstantinidis, E., & Cotronis, Y. (2013). Graphics processing unit acceleration of the red/black SOR method. Concurrency and Computation: Practice and Experience, 25(8): 1107-1120.